FAST MOTION ESTIMATION ALGORITHM IN H.264 STANDARD

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

Master ofTechnology
in
Electronic Systems and Communication

By

KALYANI MURMU
(Roll.No:211EE1119)

Under the Guidance of
Prof. DIPTI PATRA

Department of Electrical Engineering

National Institute of Technology, Rourkela

Rourkela-769008

(2011-2013)

Department of Electrical Engineering
National Institute of Technology Rourkela
Rourkela–769008, Orissa, India.

CERTIFICATE

This is to certify that the thesis entitled, "**FAST MOTION ESTIMATION ALGORITHM IN H.264 STANDARD"** submitted by **Ms. KALYANI MURMU**in partial fulfillment of the requirements for the award of Master of Technology Degree in **Electrical Engineering** with specialization in **"ELECTRONIC SYSTEMS AND COMMUNICATION"** at the National Institute of Technology, Rourkela is an authentic work carried out by her under my supervision and guidance.

To the best of my knowledge, the matter embodied in the thesis has not been submitted to any other University / Institute for the award of any Degree or Diploma.

Date:

Place:

Prof. Dipti Patra
Department of Electrical Engineering
National Institute of Technology
Rourkela-769008

ACKNOWLEDGEMENT

I have been very fortunate to have Prof. Dipti Patra, Department of Electrical Engineering, National Institute of Technology, Rourkela as myguide. She introduced me to the field of Image Processing and ComputerVision, guided me patiently throughout this thesis work. She has been a perfect motivator,very cooperative and an inspiring guide to me to fulfill this academic pursuit. Iam highly indebted and express my deep sense of gratitude to her. I am extremelythankful to her for her incredible contribution in writing the thesis.

I am grateful to professors of my specialization, Professors P. K. Sahu, S. Das, K.R. Subhasini, and S. Gupta for their reviews and care. I am also grateful to the Head of the Department of Electrical Engineering, NIT Rourkela for providing the facilities for the work. Thanksto other faculty members in the department as well.

Thanks to Mr.Yogananda, Mrs. Prajna, Ms. Smita for their reviews and constructive criticism on my work.

Kalyani Murmu

ABSTRACT

In H.264/AVC standard, the block motion estimation pattern is used to estimate the motion which is a very time consuming part. Although many fast algorithms have been proposed to reduce the huge calculation, the motion estimation time still cannot achieve the critical real time application. So to develop an algorithm which will be fast and having low complexity became a challenge in this standard.For this reasons, a lot of block motion estimation algorithms have been proposed. Typically the block motion estimation part is categorized into two parts. (1) Single pixel motion estimation (2) Fractional pixel motion estimation. In single pixel motion estimation one kind of fast motion algorithm uses fixed pattern like Three Step search, 2-D Logarithmic Search. Four Step search,Diamond Search, Hexagon Based Search. These algorithms are able to reduce the search point and get good coding quality. But the coding quality decreases when the fixed pattern does not fit the real life video sequence. In this thesis we tried to reduce the time complexity and number of search point by using an early termination method which is called adaptive threshold selection.We have used this method in three step search (TSS) and four step search and compared the performance with already existing block matching algorithm.

In order to reduce the bit rate of video signals, motion compensation prediction is applied in modern video coding technology. This is a form of temporal redundancy reduction in which the current coding frame is predicted by a motion-compensated prediction from some other already decoded frame according to motion vector. As real motion has arbitrary precision, many video coding standards allow motion vectors to have “sub-pixel” resolution. The 1/4-pel displacement vector resolution is a part of AVS (Audio Video system). In order to estimate and compensate 1/4-pel displacements, the image signal on sub-pel position has to be generated by interpolation. This increases the complexity.

This thesis work proposes fast sub-pixel motionestimation techniques having lower computational complexity.The proposed methods are based on mathematical models of the motion compensated prediction errors in compressingmoving pictures. Unlike conventional hierarchical motionestimation techniques, the proposed methods avoid sub-pixelinterpolation and subsequent secondary search after theinteger-precision motion estimation, resulting in reducedcomputational time. In order to decide the coefficients of themodels, the motion-compensated prediction errors of the neighboring pixels around the integer-pixel motion vector areutilized.

Table of Contents

		Pages
Certificate		ii
Acknowledgement		iii
Abstract		iv
List of figures		vii
List of tables		ix
1 INTRODUCTION		1
1.1	Overview	1
1.2	Various Video Compression Standard	1
1.3	Technical Overview of H.264/AVC	3
1.4	Highlights of H.264/AVC Standard	10
1.5	Motivation	11
2	MOTION ESTIMATION AND VARIOUS ALGORITHMS	12
2.1	Basics of Motion Estimation	12
2.2	Types of fast BMA	18
2.3	Search Techniques	20
2.4	Performance Study	29
2.5	Observations	32
3	EARLY TERMINATION USING THRESHOLD	33
3.1	Introduction	33
3.2	Adaptive Threshold Selection	33
3.3	Proposed Modified Three Step Search (TSS)	34
3.4	Proposed Modified Four Step Search (4SS)	36
3.5	Results and Analysis	38
3.6	Conclusion	44
4	SUB-PIXEL MOTION ESTIMATION	45
4.1	Introduction	45
4.2	Fundamentals of Sub-pixel Motion Estimation	45
4.3	HFPS Method in H.264	47
4.4	Study of Fractional Pixel Accuracy	48
4.5	Observations	52

5 FAST SUB-PIXEL MOTION ESTIMATION TECHNIQUE WITH LOWER COMPUTATIONAL COMPLEXITY 53
5.1 Introduction 53
5.2 Mathematical Models 54
5.3 Results and Analysis 59
5.4 Conclusion 62
CONCLUSION AND FUTURE WORK 63
REFERENCES 64

LIST OF FIGURES

Figures		Pages
1.1	H.264Encoder	4
1.2	H.264Decoder	5
1.3	Segmentation of macroblocksi.e.16X16 and8X8	7
1.4 Multi frame motion compensation		8
2.1 The process of matching a rectangular block of pixels in the current frame Within a search window that is larger than the rectangular block but smaller than the image		16
2.2 Three Step Search Procedure		22
2.3 New Three Step search Procedure		23
2.4 Search patterns corresponding to each selected quadrant (a) shows all quadrants (b) Quadrant I is selected (c) Quadrant II is selected (d) Quadrant III is selected (e) Quadrant IV is selected		24
2.5 Simple and Efficient Search Procedure		24
2.6 Search Pattern of FSS (a) First Step (b) Second/Third Step (c) Second/Third Step (d) Fourth Step		25
2.7 Four Step Search Procedure		26
2.8 Two different pattern of DS (a) LDSP (b) SDSP		27
2.9 Diamond Search Procedure		27
2.10 Adaptive Rood Pattern		28
2.11 Figure Showing the plot of search point per macroblock at various frame number for different BMA i.e. ARPS, DS, ES,NTSS,SESTSS,SS4, TSS for Caltrain sequence		29
2.12 Figure showing the plot PSNR at various frames number for different BMA i.e. ARPS, DS, ES, NTSS, SESTSS,SS4, TSS for Caltrain sequence		30
2.13 Figure Showing the plot of search point per macroblock at various frame number for different BMA i.e. ARPS, DS, ES,NTSS,SESTSS,SS4, TSS for Salesman sequence		30
2.14 Figure showing the plot PSNR at various frame number for different BMA i.e. ARPS, DS, ES, NTSS, SESTSS, SS4, TSS for Salesman sequence		31

2.15 Figure Showing the plot of search point per macroblock at various frame number for different BMA i.e. ARPS, DS, ES,NTSS,SESTSS,SS4, TSS for Missa sequence 31
2.16 Figure showing the plot PSNR at various frame number for different BMA i.e. ARPS, DS, ES, NTSS, SESTSS,SS4, TSS for Missa sequence 32
3.1 Flowchart of Improved TSS using threshold 35
3.2 Flowchart of Improved 4SS using threshold 37
3.3 Figure Showing the plot of search point per macroblock at various frame number for different BMA i.e. TSS, NTS,SS4,SS4TH for Caltrain sequence 39
3.4 Figure Showing the plot of search point per macroblock at various frame number for different BMA i.e. TSS, NTS,SS4,SS4TH for Salesman sequence 39
3.5 Figure Showing the plot of search point per macroblock at various frame number for different BMA i.e. TSS, NTS,SS4,SS4TH for Missa sequence 40
3.6 Figure showing the plot of PSNR at various frame number for different BMA i.e. TSS, NTSS,SS4,SS4TH for caltrain sequence 40
3.7 Figure showing the plot of PSNR at various frame number for different BMA i.e. TSS, NTS,SS4,SS4TH for salesman sequence 41
3.8 Figure showing the plot of PSNR at various frame number for different BMA i.e. TSS, NTS,SS4,SS4TH for missa sequence 41
3.9 Sample of sequences used for the study of algorithms 42
4.1 HFPS pattern 47
4.2 Two frames of Table tennis sequence 48
4.3 Motion Vector at different accuracy i.e. single pixel, Half pixel, Quarter pixel, 1/8 pixel accuracy 49
4.4 Motion Compensated error image at different accuracy i.e. single pixel, Half pixel, Quarter pixel, 1/8 pixel accuracy 50
5.1 Position of integer pixel and half pixel 54
5.2 Position of integer pixel, half pixel and quarter pixel 56
5.3 Sample of sequences used in the model 66

LIST OF TABLES

Tables	Pages
3.1 Evaluation of Avg. no. search point and ME Time for 4 different sequences (1) caltrain (2) Akiyo (3) Foreman (4) Table tennis	38
3.2 MSE Performance of 4 different sequences (1) Caltrain (2) Akiyo (3) Foreman (4) Table tennis	43
4.1 Performance of different fractional pixel accuracy for different block size By measuring MSE with MC and MSE without MC for Table tennis Sequence	51
4.2 Performance of different fractional pixel accuracy for different block size By measuring MSE with MC and MSE without MC for Foreman Sequence	51
4.3 Performance of different fractional pixel accuracy for different block size By measuring MSE with MC and MSE without MC for Foreman Sequence	52
5.1 The Values of MSE with MC for Model-1 and Model-2 at different Block sizes for 5 standard sequences (1) Akiyo (2) Foreman (3) Salesman (4) Caltrain and (5) Table tennis	59
5.2 Motion Estimation Time in Second in HFPS, Model-1 and Model-2 for 5 standard sequences (1) Akiyo (2) Foreman (3) Salesman (4) Caltrain(5) table tennis	62

CHAPTER-1

Introduction

1.1Overview:

The increasing demand to incorporate video data into telecommunications services, the corporate environment, the entertainment industry, and even at home has made digital video technology a necessity. A problem, however, is that still image and digital video data rates are very large, typically in the range of 150Mbits/sec. Data rates of this magnitude would consume a lot of the bandwidth, storage and computing resources in the typical personal computer. For this reason, Video Compression standards have been developed to eliminate picture redundancy, allowing video information to be transmitted and stored in a compact and efficient manner.

1.2 Various Video Compression Standards:

There are two types of compression lossy and lossless. Lossless data compression is used when the data must be restored exactly as it was before compression.Lossy compression is used when the data doesn't have to be restoredperfectly. Two main standards bodies are working parallely for the development of video compression standards, the Moving Picture Experts Group (MPEG) and the International Telecommunication Union (ITU). Five video compression standards[2] are discussed below briefly.

1.2.1 MPEG-1:

MPEG-1, the first lossy compression scheme developed by the MPEG committee, is now a days used for CD-ROM video compression and as part of previous Windows Media players. TheMPEG-1 algorithm uses Discrete Cosine Transform (DCT) algorithm which initially convert each image into the frequency domain and then process these in frequency coefficients to optimally reduce a videostream to the required bandwidth. The DCT algorithm is well known and widely used for data compression. Similar to the FastFourier Transform, DCT converts data, such as the pixels in an image into sets offrequencies.The current wildly popular MP3 (MPEG-1, Part 3) audio standardis actually the audio compression portion of the MPEG-1 standard and provides about 10:1compression of audio files at reasonable quality [2].

1.2.2 MPEG-2:

The MPEG-2 compression standard is used to overcome the requirements of compressing higher quality videos. It is the combination of lossy video compression and lossy audio data compression methods, which is used to store and transmit pictures using currently available

storage media and transmission bandwidth. MPEG-2 is generally used as the format of digital television signals that are broadcast by terrestrial, cable and direct broadcast satellite TV systems. It uses bit rates typically in the range from 5 to 8 Mbits/s, although MPEG-2 is not necessarily limited to a bit rate range. MPEG-2's basic compression techniques are very similar to MPEG-1 by using DCT transforms but it also gives support for interlaced video (the format used by broadcast TV systems) [2].

1.2.3 MPEG-3:

The MPEG committee originally intended that an MPEG-3 standard would evolve to support HDTV,but it turned out that this could be done with minor changes to MPEG-2. So MPEG-3 never happened, and now there are profiles of MPEG-2 that support High Definition Television (HDTV) as well as Standard Definition Television (SDTV) [2].

1.2.4MPEG-4:

MPEG-4 is a method of defining compression of audio and visual (AV) digital data. It is used in the compression AV (audio and visual) data for web and CD distribution voice and broadcast television application. The main aim of the MPEG-4 standard is to find solution for two video transport problems: sending video over low bandwidth channels such as the video cell phones and internet, and achieving better compression compared to MPEG-2 for broadcast signals.MPEG-4 performs well in terms of compression and it is used in the range of 64 Kbits/s to 1,800 Mbits/s. However, it has limited success in achieving better compression than MPEG-2 for broadcast signals although it is in the range of15% more than MPEG-2, this has not been enough of an advantage to convert the whole broadcast industry to MPEG-4. So, MPEG-4 is mostly used in lower-bandwidth applications, like desktop cell phone, Internet and desktop computer [2].

1.2.5 H.264/AVC:

As MPEG-4 fails to improve compression performance for full broadcast signals, the H.264 is developed to achieve a improvement of 2:1 over MPEG-2 on full-quality SDTV and HDTV.H.264/MPEG-4 AVC is a block oriented motion compensation based codec standard jointly developed by theISO/IEC Moving Picture Expert Group (MPEG) and ITU-T Video Coding Expert Group (VCEG). The project partnership effort is known as the Joint Video Team (JVT). It is also known as MPEG-4 part 10 AVC (Advanced Video Compression).The main aim of the H.264/MPEG-4 AVC is to provide significantly improved compression performance and provision for a network-friendly packet-based video representation addressing conversational (video telephony) and non-conversational (storage, broadcast or streaming) applications. H.264

technique is different from MPEG-2 and can match the MPEG-2 quality at up to half the data rate. H.264 can deliver excellent video quality across the entire bandwidth spectrum from 3G to HDTV and many more in between 40 Kbits/s to upwards of 10 Mbits/s. H.264 standard contains a lot of features that allows it to compress video much more effectively than older codecs [2].

1.2.6 JPEG 2000:

The MPEG and H.264 standards all related to motion video. For still pictures, the familiar JPEG standard is developed by the Joint Picture and Editors Group Committee has been in use for some years, and it is just now gradually being replaced by the JPEG committee's improved JPEG 2000 standard, which was released in 2000.Even though JPEG 2000 isdesigned for still picture, Part 3 of the JPEG 2000 standard Motion JPEG 2000 is used for motion video. JPEG 2000 uses wavelet compression technology instead of DCT technology used in theMPEG and JPEG standards. DCT compresses an image into 8x8 pixel blocks and places them consecutively in the file. The blocks are compressed individually, without reference to the adjoining blocks, resulting in the blocky look associated with compressed JPEG files. JPEG 2000 is able to render pictures better by eliminating the blockinessthat is a common feature of DCT compression. It also producessmaller image files than JPEG image files with the same level of compression [2].

1.3 Technical Overview of H.264/AVC:

The H.264/AVC standard [1] offers (1) broadcast over satellite, cable modem and so on, (2) interactive or serial storage on optical or magnetic recording devices such as Blue Ray DVD, (3) conversational services over LAN, Ethernet, wireless and mobile networks, (4) video on demand or multimedia streaming services over cable modem, DSL, and so on, and (5) multimedia messaging services over DSL and ISDN, mobile and wireless networks, and so on. The H.264/AVC standard covers two systems, named as efficient video coding layer (VCL) for the compression of video data and a network abstraction layer (NAL), which converts the VCL data to a suitable manner for a variety of service media to carry the compressed data [2].

1.3.1 Network Abstraction Layer (NAL):

NAL allows same video syntax to be used in more than one network environments. The NAL is specified to convert the VCL data and gives header information in a manner suitable for the transport layer or storage media. All data are contained in NAL units, each of which contains an integer number of bytes. The NAL unit specifies a generic format for use in both packet oriented and bitstream of systems [2].

1.3.2 Video Coding Layer (VCL):

The video coding layer of H.264/AVC is same as other standards like MPEG-2 Video. It consists of a hybrid of temporal and spatial prediction, in addition with transform coding. Figure 1.1 shows a block diagram of the video coding layer for a frame**.**

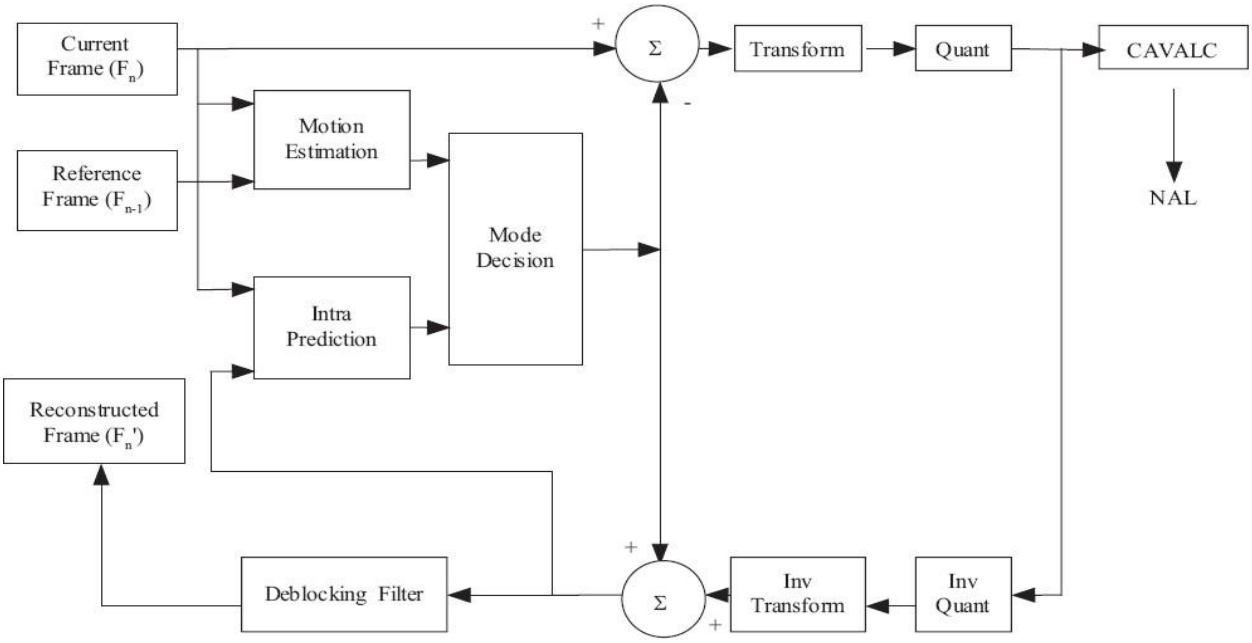


Figure 1.1:H.264 Encoder

In general with earlier video coding standards, H.264 does not explicitly define a CODEC [7] (enCOder/DECoder pair) but it rather defines the syntax of an encoded video bitstream together with decoding this video bitstream. In practice, a compliant encoder and decoder are likely to include the functional elements shown in Figure 1.1 and Figure 1.2. Excluding the deblocking filter, most of the basic functional elements (prediction, transform, quantization, entropy encoding) are present in previous standards (MPEG-1, MPEG-2, MPEG-4, H.261, H.263) but the important changes in H.264 occur in the details of each functional block.

The encoder (Figure 1.1) having two dataflow paths, a 'forward' path (left to right) and a 'reconstruction' path (right to left). The dataflow path in the decoder (Figure 1.2) is shown from right to left to explain the similarities between Encoder and Decoder. The following description is simplified in order to provide an overview for method of encoding and decoding. The term 'block' is used to denote a macroblock partition or sub-macroblock partition (inter coding) or a 16 x 16 or a 4 x 4 block of luma samples and associated chroma samples (intra coding).

An input frame F_n is used as input in units of a macroblock. Each macroblock is encoded in intra mode or inter mode and for each block in the macroblock, a prediction PRED is formed based on the reconstructed picture samples. In intra mode, PRED is formed from the samples in the current slice that are previously encoded, decoded and reconstructed (uF_n' in the Figure 1.2). In inter mode, PRED is formed from motion compensated prediction by one or two reference frames selected from the set of list 0 and/or list 1 reference frames. In the figure the reference picture is shown as the previous encoded picture F'_{n-1}but the prediction reference for each macroblock partition (in inter mode) may be selected from a list of past or future pictures (in display order) that has already been encoded, reconstructed and filtered.

The prediction PRED is subtracted from the current block to produce a differential (difference)block D_n, which is transformed (using a block transform) and quantized to give X, and a set of quantized transform coefficients which are reordered and entropy encoded are formed. The entropy encoded coefficients, together with side information are required to decode each block within the macroblock (prediction modes, quantiser parameter, motion vector information, etc.)from the compressed bitstream which is passed through Network Abstraction Layer(NAL) for storage and transmission [7].

As well as encoding and transmitting each block in a macroblock, the encoder decodes (reconstructs) it to provide a reference for further predictions. The coefficients X are inverse quantized and inverse transformed to produce a difference blockD_n'. The prediction block PRED is added to D_n' to create a reconstructed block uF_n' (a decoded version of the original block; u indicates that it is unfiltered). A filter is applied to reduce the effects of blocking distortion and the reconstructed reference picture is created from a series of blocks F_n'.

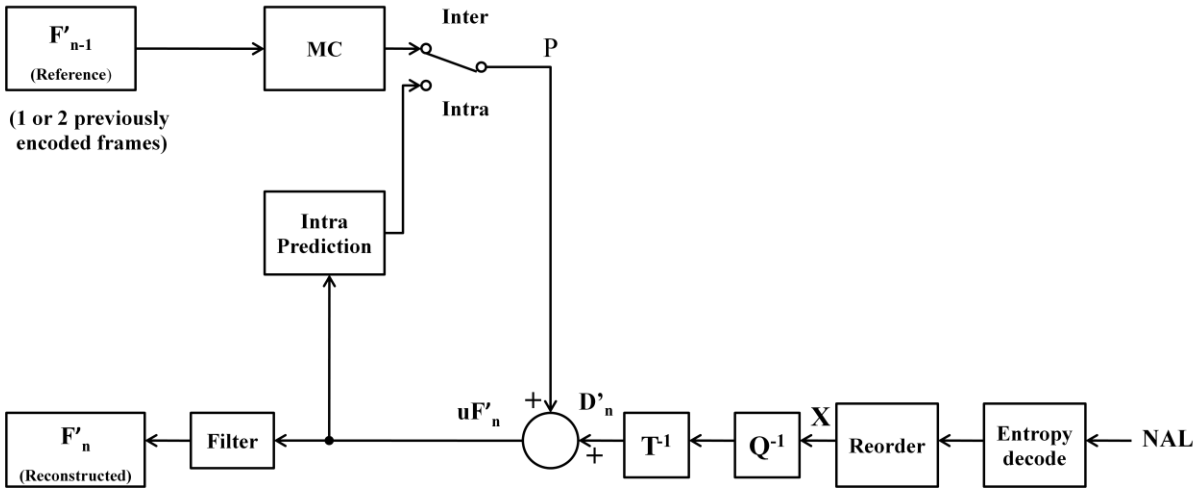


Figure 1.2:H.264 decoder

The decoder receives a compressed bitstream from the NAL and entropy decodes the data elements to produce a set of quantized coefficients X. These are scaled and inverse transformed to give D_n' (identical to the D_n' shown in the encoder). Using the header information decoded from the bitstream, the decoder creates a prediction block PRED, identical to the original prediction PRED formed in the encoder. PRED is added to D_n' to produce uF_n' which is filtered to create each decoded block F_n' [7].

1.3.3 Subdivision of Pictures into Macroblocks:

Each picture of a video, which can either be a frame or a field, is again into fixed-size macroblocks that cover a rectangular picture area of 16×16 samples of the luma component and 8×8 samples of each of the twochroma components. All luma and chroma samples are either spatially or temporally predicted, and the resulting prediction is transmitted using transform coding. So, each colorcomponent of the prediction residual is again divided into blocks. Each block is transformed using an integer transform, and the transform coefficients are quantized and transmitted using entropy-coding methods.

The macroblocks are organized in slices, which is a subset of pictures that can be decoded independently.H.264/AVC standard supports five different slice coding types.

1. I Slice ('I' stands for Intra)
2. P Slice ('P' stands for Predictive)
3. B Slice ('B' stands for Bi-predictive)
4. SP Slice (Switching P)
5. SI Slice (Switching I)

In 'I' slice all macroblock are coded without referring to other pictures within the video sequences but in case of P and B slices a prior coded image can be used to form a prediction signal. The remaining two slices SP and SI are specified for efficient switching between bitstream coded at various bit rates. The Inter prediction signals of the bitstreams for one selected SP frame are quantized in the transform domain, forcing them into a coarser range of amplitudes.SI frames are used to achieve a perfect match for SP frames when Inter prediction cannot be used because of transmission errors [1].

1.3.4 Intra Frame Prediction:

Each macroblock can be transmitted in one of several coding types depending upon the slice-coding type. In every slice-coding types, two classes of intra coding types are supported.(1)INTRA-4X4 (2) INTRA-16X16. In the previous standards where prediction is

conductedinthe transform domain, in H.264/AVC prediction is always conducted in the spatial domain by referring to neighboring samples of already coded blocks.

In INTRA-4X4 mode, each 4X4 block of luma component utilizes one of nine prediction mode. The nine prediction modes are mode 0 (horizontal prediction), mode 2 (DC prediction), mode 3 (diagonal down/left prediction),mode 4 (diagonal down/right prediction),mode 5 (vertical-right prediction),mode 6 (horizontal-down prediction),mode 7 (vertical-left prediction),and mode 8 (horizontal-up prediction). In INTRA-16X16 modes, a uniform prediction is performed for the whole luma component of a macroblock, which is suited for smooth area of images.

1.3.5 Motion Compensation in P Slices:

Each P-type macroblock corresponds to a specific partitioning of fixed-size blocks which used for motion description. Partitions with luma block sizes of 16X16, 16X8, 8X16 and 8X8 samples are supported by the Inter-16×16, Inter-16X8, Inter-8X16 and Inter-8X8 macroblock types, respectively.

Figure 1.3 explains the partition. The prediction signal for each mxnluma block is obtained by displacing an area of the corresponding reference picture, which is specified by a translational motion vector and a picture reference index. So, if the macroblock is coded using the Inter-8x8 macroblock type, and each sub-macroblock is coded using the Inter-4x4 sub-macroblock type, a maximum of sixteen motion vectors may be transmitted for a single P-slice macroblock [4].

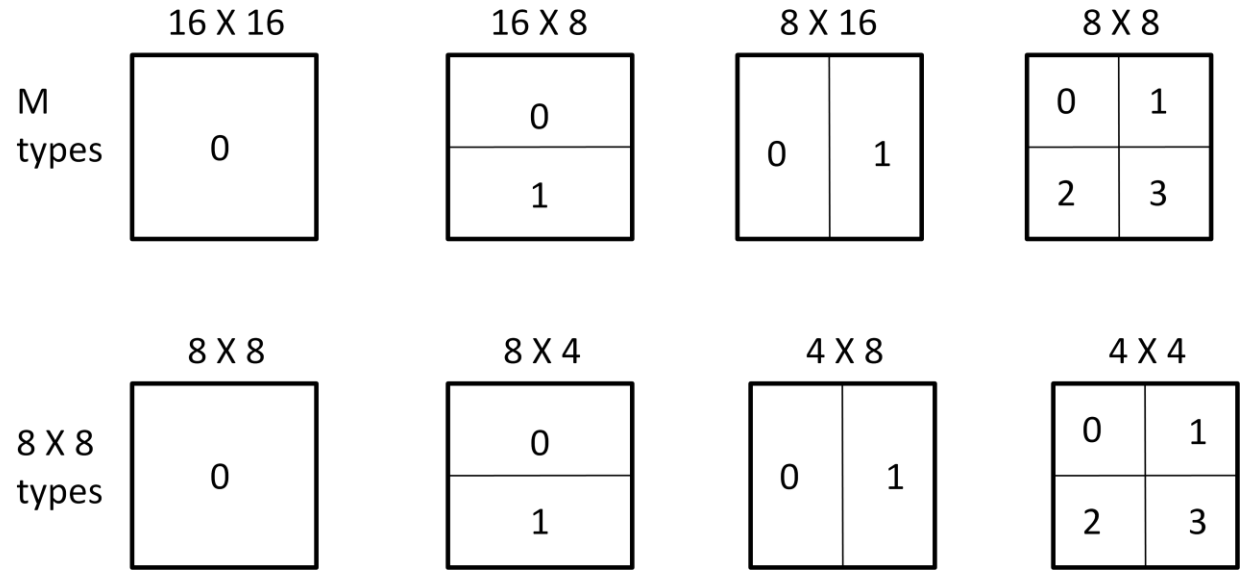


Figure 1.3: Segmentation of macroblock for motion compensation

Top- Segmentation of macroblock

Bottom-Segmentation of 8 X 8 block

H.264/AVC supports multi-picture motion-compensated prediction. That is, more than one prior-coded picture can be used as a reference for motion-compensated prediction.Figure 1.4 explains the concept.

Both the encoder and decoder have to store the reference pictures used for Inter-prediction. The decoder replicates the multi-picture buffer of the encoder. Until the size of the multi-picture buffer is set to one picture, the index at which the reference picture is located inside the multi-picture buffer has to be signaled. The reference index parameter is transmitted for each motion-compensated 16X16, 16X8, 8X16 or 8X8 luma block.

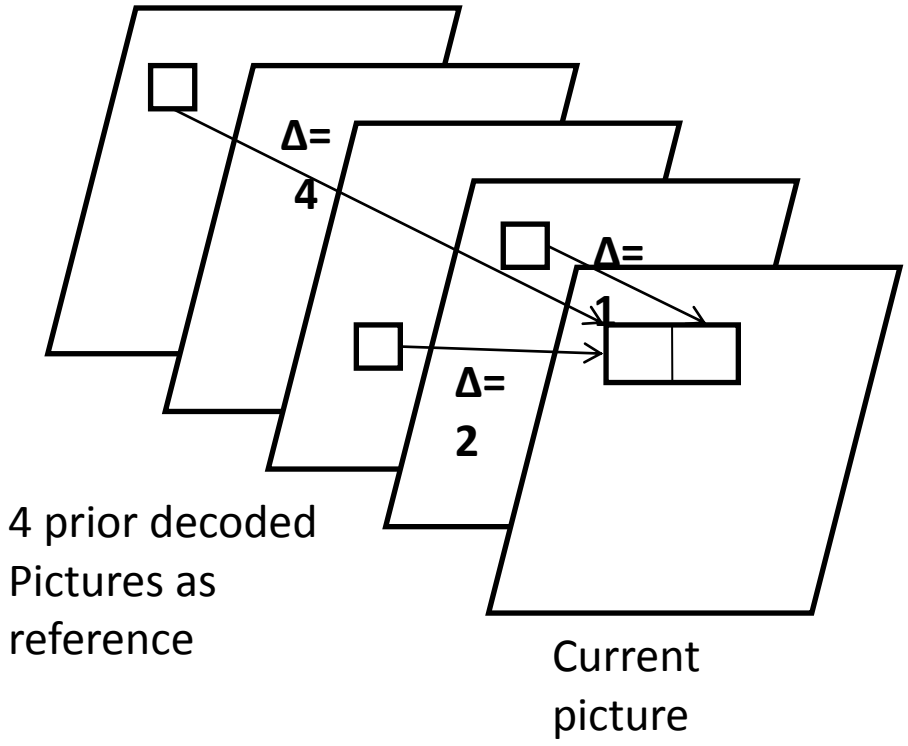


Figure 1.4: Multi frame motion compensation.
Motion vector and picture reference parameter (Δ)
is transmitted

1.3.6 Motion Compensation in B Slice:

In comparison to previous video-coding standards, the concept of B slices is generalized in H.264/AVC. The basic difference between Band P slices is that B slices are coded in a manner in which some macroblock or blocks may use a weighted average of two different motion-compensated prediction values, for making the prediction signal. In B slices, four different types of inter-picture prediction are supported: list 0, list 1, bi-predictive, and directprediction. The list

0 prediction indicates that the prediction signal is formed by utilizing motion compensationfrom a picture of the first reference picture buffer. If list 1 prediction is used a picture of the second reference picture buffer is used for building the prediction signal. In the bi-predictive mode, the prediction signal is formed by a weighted average of a motion-compensated list 0 and list 1 prediction signal. The direct prediction mode is derived from previouslytransmitted syntax elements and can be either list 0 or list 1 prediction or bi-predictive.

B slices uses a similar macroblock partitioning as P slices. Instead of using Inter-16X16, Inter-16X8, Inter-8X16,Inter-8X8 and the Intra modes, a macroblock utilizes direct prediction, i.e. the direct mode, is provided. For each 16X16, 16X8, 8X16 and 8X8 partition, the prediction method (list 0, list1, bi-predictive) can be chosen separately. An 8X8 partition of a B-slice macroblock can also be coded in direct mode. The motion vector coding is similar to that of P slices with the appropriate modifications because neighboring blocks may be coded using different prediction modes.

1.3.7 Transform, scaling and quantization:

Similar to prior video coding standards, H.264/AVC also utilizes transform coding for the prediction residual. However, in H.264/AVC, the transformation is applied for4X4 blocks, instead of a 4X4 discrete cosine transform (DCT), a separable integer transform which have basically the same properties as 4X4DCT is used. An extra 2X2 transform is applied to the four DC coefficients of each chroma component. If a macroblock is coded in Intra-16x16 modes, a similar 4x4 transform is performed for the 4x4 DC coefficients of the luma signal. H.264/AVC uses scalar quantization for the quantization of transform coefficients. One among 52 quantizers is selected for each macroblock by the Quantization Parameter (QP). The quantizers are arranged so that thereis an increase of approximately 12.5% in the quantization step size when the QP is increasing by one. Thequantized transform coefficients of a block are generally scanned in a zigzag manner and transmitted using entropy coding methods. The 2X2DC coefficients of the chroma component are scanned in raster-scan order.

1.3.8 Entropy Coding:

There are two methods of entropy coding that supports H.264/AVC. The default entropy coding method employs a single infinite-extend codeword set for all syntax elements, except the quantized transform coefficients. For transmitting the quantized transform coefficients, a sophisticated method called Context Adaptive Variable Length Coding (CAVLC) is used. In this scheme, VLC tables for various syntax elements are available, depending on already transmitted syntax elements. Since the VLC tables are well designed for matching the corresponding

conditioned statistics, the entropy coding performance is improved in comparison to schemes using VLC table. The efficiency of entropy coding can be improved further if Context-Adaptive Binary Arithmetic Coding (CABAC) is employed.

1.3.9 In Loop Deblocking Filter:

One of the main characteristic of block-based coding is visible block structures. Block edges are typicallyreconstructed with less accuracy than internal pixels and "blocking" is generally considered to be one of themost visible defect with the present compression methods. For this reason H.264/AVC employs an adaptivein loop deblocking filter, where the strength of filtering is controlled by varying the values of syntax elements. The blockiness is reduced without much affecting the sharpness of the content. In this manner the quality of subjective part is improved.

1.4Highlight of H.264/AVC Feature:

As here we concern for video compression, we will briefly highlight some features of H.264/AVC [5] to get a better perspective of how compression efficiency is improved over MPEG-2.

- Variable Block-size Motion Estimation*:* Motion compensation can be performed on blocks of 4X4 pixels instead of fixed size macroblock.
- Quarter-pel prediction*:* To reduce the prediction residuals the accuracy is increased in quarter pel resolution in the process of motion estimation.
- Motion Vectors over picture boundaries*:* Picture boundary extrapolations are used to allow motion vectors to point out the areas outside the previously reconstructed reference picture.
- Independent decoding and display order*:* H.264/AVC allows the encoder to choose the ordering of pictures and displaying with a high degree complexity.
- Motion compensation using multiple reference pictures.
- Flexibility in B-picture coding.
- Weighted prediction*:* Motion compensated prediction residuals can be weighted and offset to improve coding efficiency.
- Improved intraprediction*:* Use of directional prediction improves coding efficiency.
- In-loop deblocking filter.
- 4X4 transform.
- Hierarchical block transform.
- Short word length for transform.

- Exact match in inverse transform.
- Advanced arithmetic coding*:* H.264 uses two advanced arithmetic coding methods known as CABAC (context-adaptive binary arithmetic coding), and CAVLC (context-adaptive variable length coding) to improve the performance from MPEG-2.

1.5 Motivation:

The H.264/MPEG-4 Advanced Video Coding standard (H.264/AVC) [1] is the newest video coding standard jointly developed by the ITU-T Video Coding Experts Group (VCEG) and the ISO/IEC Moving Picture Experts Group(MPEG). H.264/AVC has achieved great improvement in compression performancecompared to existing standards. It provides a network friendly representation of the video data which is suitable for both conversational (video telephony) and non-conversational (storage, broadcast, or streaming) applications.

The increasing demand of video services and the growing popularity of higher definition video are constantly making demand for improved compression capabilities and this in turn motivated the H.264/AVC standard to improve their coding efficiency, architecture and functionalities.So that the overall performance can be improved.

In H.264/AVC standard the block motion estimation pattern is used to estimate the motion where time complexity plays an important role in video coding. Although many fast algorithms have been proposed to reduce the huge computation, the motion estimation time still cannot achieve the critical real time application. So to develop an algorithm which will be fast and having low complexity became a challenge in this standard.

For the above reasons, a lot of block motion estimation algorithms have been proposed. Typically the block motion estimation part is categorized into two parts. (1) Single pixel motion estimation (2) Fractional pixel motion estimation. In single pixel motion estimation one kind of fast motion algorithm uses fixed pattern like Three Step search[8],New Three Step Search [9],Four Step search[11], Block Based Gradient Descent search[10], Diamond Search[12],Hexagon Based Search[13].These algorithms are able to reduce the search point and get good coding quality. But the coding quality decreases when the fixed pattern does not fit the real life video sequence. So some hybridized fast motion estimation algorithms are proposed to achieve high speed and accuracy. In fractional pixel motion estimation the PSNR is improved about 2-3dB as compared to integer pixel motion estimation and also has high computation complexity due to complex sub-pels interpolation process.

CHAPTER – 2

Motion Estimation and Various Algorithms

2.1Basics of Motion Estimation

Motion estimation is the process of determining motion vector that illustrates the transformation from one 2D image to another; usually from neighboring frames in a video sequence. The motion vectors may be related to the complete image (global motion estimation) or specific parts, such as rectangular blocks, arbitrary shaped patches or even per pixel. The motion vectors may be represented by a translational model or many other models that can approximate the motion of a real video camera, such as rotation and translation in all three dimensions and zoom.

An object's motion can be estimated using variations in pixel values. These pixel variations may be due to lighting conditions, electronic noise or apparent object motion.Since the 2D image is a perspective projection of 3D object, it's difficult to know whether the motion is due to rigid body or deformable body. A deformable body motion means that different parts of the same body undergo motion in different extent.

In order to determine the motion of an object in a frame, one has to first define the object boundaries in the current and reference frames. To make it simple, we will only use rectangular blocks of a constant size. The initial task is to estimate the translational motion of a rectangular block of pixels between the current and previous or reference frames known as the motion estimation. This implies a motion vector with components in the horizontal and vertical directions. These parts can be positive or negative. Apositive value means motion to the right or motion to the downward and negative value means motion to the left or motion to the upward. This motion vector will be used to predict new frame from the reference which is called motion compensation. The magnitude of the components will be in number of pixels. Once the motion vector is known, the rectangular block in the current frame can be aligned with that in the reference frame and the corresponding differential pixels can be found. The process of aligning and differencing objects between successive frames is called motion-compensated (MC) prediction [4]. As we are dealing with images defined on rectangular grids, motion can only be estimated to an accuracy of a pixel. But it is also possible to estimate the motion to an accuracy of less than a pixel only approximately because it involves interpolating the values between pixels which is known as sub-pixel motion estimation or fractional pixel motion estimation.

Our purpose is to estimate the translational motion of a block of pixels in the current frame with respect to the previous or reference frame. This can be achieved by three methods [5] which are described below:

(1) Phase Correlation Method or Pel Recursive Method

(2) Optical Flow Equation

(3) Block Matching Method

2.1.1 Phase Correlation Method:

This method is applied on the assumption that the normalized cross-correlation function in the two dimensional (2D)Fourier domain estimates the relative phase shift between two blocks of the image.

Let us assume that $b[m,n,k]$and $b[m,n,k-1]$are the rectangular blocks in the current frame and previous frame respectively. Then the cross-correlation between the two blocks is determined by the convolution of the individual blocks and is given by,

$$c_{k,k-1}[m,n] = b[m,n,k] \otimes\otimes b[m,n.k-1] \tag{2.1}$$

The symbol $\otimes\otimes$ denotes 2D convolution. The 2D Fourier transform of equation (2.1) gives the complexed-valued cross-power spectrum. Thus,

$$c_{k,k-1}(u,v) = B_k(u,v)B^*_{k-1}(u,v) \tag{2.2}$$

Where $B_j(u,v), j = k-1, k$ is the 2D Fourier Transform of is $b[m,n.j]$,u is the Fourier frequency in the vertical direction and v is the Fourier frequency in horizontal direction. * denotes complex conjugate. If the motion between the blocks d_x in the horizontal direction and d_y in the vertical direction, then the two Fourier Transform are related by,

$$B_k(u,v) = B_{k-1}(u,v)\exp\left(j\left(ud_y + vd_x\right)\right) \tag{2.3}$$

By normalizing equation (2.3) by the magnitude of $C_{k,k-1}(u,v)$, we obtained

$$\hat{C}_{\text{k,k-1}}(u,v) = \frac{B_{k-1}(u,v)B^*_{k-1}(u,v)}{\left|B_{k-1}(u,v)B^*_{k-1}(u,v)\right|} = \exp\left(j(ud_y + vd_x)\right) \tag{2.4}$$

The 2D inverse Fourier transform of the above equation is the cross-correlation in the spatial domain and found to be,

$\hat{C}_{k,k-1}(m,n)=\delta[m-d_{y,}n-d_x]$ (2.5)

The equation (2.5) corresponds to an impulse in the 2D plane located at$[m-d_x,n-d_y]$,which is the required motion vector.

In practice the phase-correlation method is not used for motion estimation as it involves high computational complexity and phase ambiguity.

2.1.2 Optical Flow Method

Optical flow is the pattern of apparent motion of objects, surfaces, and edges in a visual scene caused by the relative motion between an observer and the scene. It can be used to estimate the motion vectors in a video sequence. If the intensity $f(x,y,t)$ is constant along a motion trajectory, then the total derivative with respect to t is t zero, that is,

$\frac{df(x,y,t)}{dt}=0$ (2.6)

By the chain rule of differentiation and putting $X=[x\ \ y]^T$

Equation (2.6) becomes

$$\frac{df(X,t)}{dt}=\frac{df(X,t)}{dx}\frac{dx}{dt}+\frac{df(X,t)}{dy}\frac{dy}{dt}+\frac{df(X,t)}{dt}=0 \quad (2.7)$$

Substituting $v_x(t)=\frac{dx}{dt}$ and $v_y(t)=\frac{dy}{dt}$

$$\frac{\partial f(X,t)}{\partial x}v_x(t)+\frac{\partial f(X,t)}{\partial y}v_y(t)+\frac{\partial f(X,t)}{\partial t}=0 \quad (2.8)$$

Equation (2.8) is called the OFE or optical flow constraints [5]. The objective is to estimate the motion,

As equation (2.8) has two variables, the solution will not be unique. So we need to minimize the mean square error of the optical flow constraint over a block of pixels, assuming the motion vector is constant over the block.

Let $D = \sum_{x\in B}\left\{\frac{\partial f(X,t)}{\partial x}v_x(\mathrm{t}) + \frac{\partial f(X,t)}{\partial y}v_y(\mathrm{t}) + \frac{\partial f(X,t)}{\partial t}\right\}^2$ (2.9)

Setting the partial derivative to zero.

$$\frac{\partial D}{\partial v_x} = \sum_{x\in B}\left\{\frac{\partial f(X,t)}{\partial x}v_x(\mathrm{t}) + \frac{\partial f(X,t)}{\partial y}v_y(\mathrm{t}) + \frac{\partial f}{\partial t}\right\}\left\{\frac{\partial f}{\partial x}\right\} = 0 \quad (2.10)$$

$$\frac{\partial D}{\partial v_y} = \sum_{x\in B}\left\{\frac{\partial f(X,t)}{\partial x}v_x(\mathrm{t}) + \frac{\partial f(X,t)}{\partial y}v_y(\mathrm{t}) + \frac{\partial f}{\partial t}\right\}\left\{\frac{\partial f}{\partial y}\right\} = 0 \quad (2.11)$$

$$\begin{bmatrix} \sum_{x\in B}\left(\frac{\partial f}{\partial x}\right)^2 & \sum_{x\in B}\frac{\partial f}{\partial y}\frac{\partial f}{\partial x} \\ \sum_{x\in B}\frac{\partial f}{\partial x}\frac{\partial f}{\partial y} & \sum_{x\in B}\left(\frac{\partial f}{\partial y}\right)^2 \end{bmatrix}\begin{bmatrix} v_x(t) \\ v_y(t) \end{bmatrix} = \begin{bmatrix} -\frac{\partial f}{\partial x}\frac{\partial f}{\partial t} \\ -\frac{\partial f}{\partial y}\frac{\partial f}{\partial t} \end{bmatrix} \quad (2.12)$$

Since we are dealing with images defined on integer pixel, the gradient can be approximated using finite difference as given by,

$$\frac{\partial f(X,t)}{\partial x} \approx \frac{1}{4}\begin{Bmatrix} f[m+1,n,k] - f[m,n,k] + f[m+1,n+1,k] \\ -f[m,n+1,k] + f[m+1,n,k+1] - f[m,n,k+1] \\ +f[m+1,n+1,k+1] - f[m,n+1,k+1] \end{Bmatrix} \quad (2.13)$$

$$\frac{\partial f(X,t)}{\partial y} \approx \frac{1}{4}\begin{Bmatrix} f[m,n+1,k] - f[m,n,k] + f[m+1,n+1,k] \\ -f[m+1,n,k] + f[m,n+1,k+1] - f[m,n,k+1] \\ +f[m+1,n+1,k+1] - f[m+1,n,k+1] \end{Bmatrix} \quad (2.14)$$

$$\frac{\partial f(X,t)}{\partial t} \approx \frac{1}{4}\begin{Bmatrix} f[m,n,k+1] - f[m,n,k] + f[m+1,n+1,k] \\ -f[m+1,n,k] + f[m,n+1,k+1] - f[m,n,k+1] \\ +f[m+1,n+1,k+1] - f[m+1,n,k+1] \end{Bmatrix} \quad (2.15)$$

2.1.3 Block Matching Method

Block matching (BM) motion estimation plays a very important role in video coding. In this approach, image frames in a video sequence are divided into blocks. For each block in the current frame, the best matching block is identified inside a region in the previous frame. It is a spatial domain process where a block of pixels in the current frame is matched to same size of block of pixels in the reference frame using suitable matching criterion. As the matching process is computationally intensive and the motion is not significant between the adjacent frames, the

matching process is limited to a search window which is much smaller than the size of the image frame. The basic idea is illustrated in the Fig2.1.

If the block size is assumed to be M X N then the window size will be double the size of the block. There are several matching criterion for estimating the motion few of them described below.

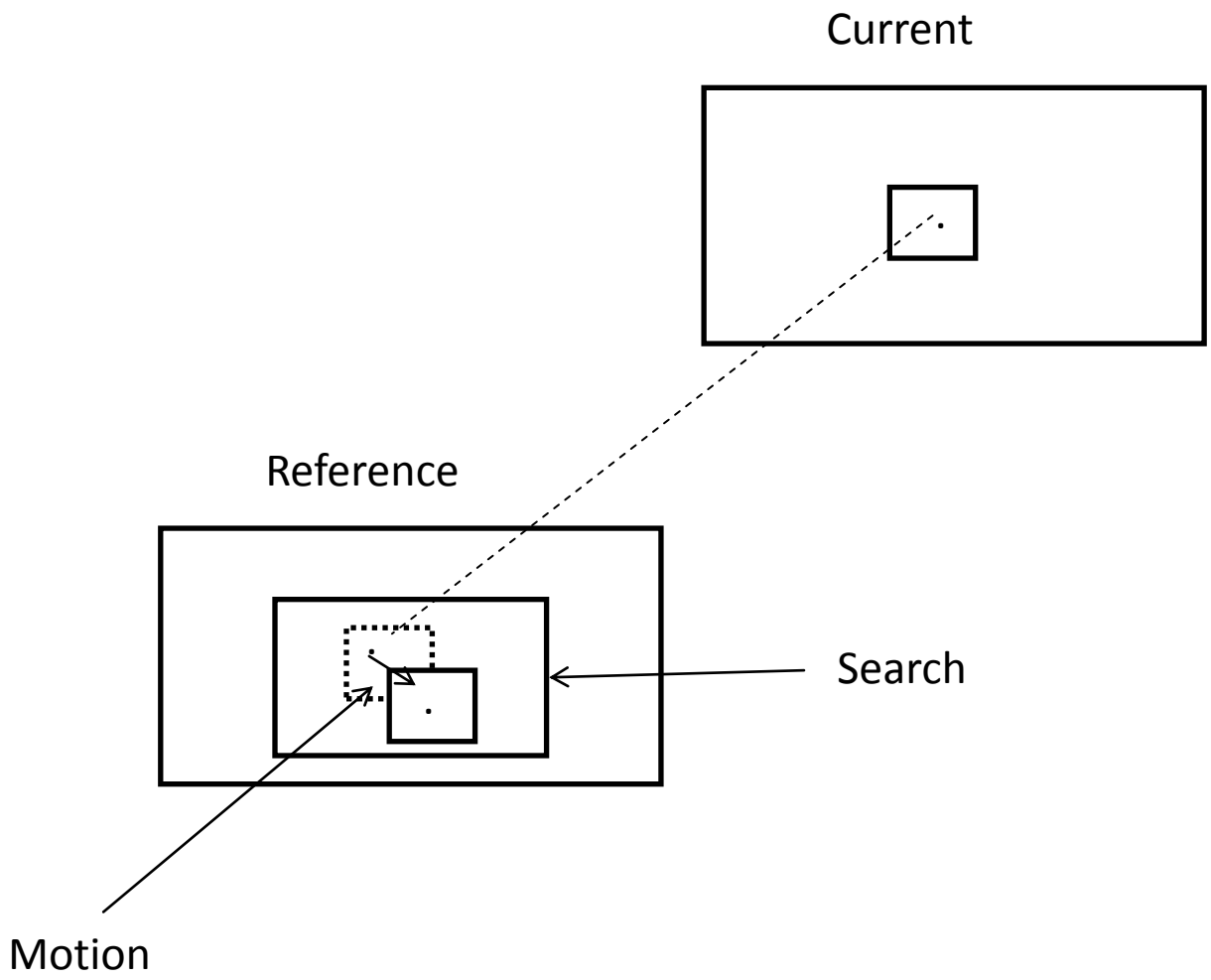


Figure 2.1: The process of matching a rectangular block of pixels in the current frame to that in the reference frame within a search window that is larger than the rectangular block but smaller than the image

2.1.3.1 MSE matching criterion:

In this case the best matching block is determined for which the MSE between the block in the current frame and the block within the search window W in the reference frame is minimum. The MSE between the blocks with the candidate displacement $dx$and$dy$is defined as,

$$\text{MSE}\ (d_{x,}d_y) = \frac{1}{M_1 N_1}\sum_{(m,n)\in W}(b[m,n,k] - b[m-dy, n-dx, k-1])^2 \quad (2.16)$$

M_1XN_1 is the size of the block

2.1.3.2 Mean Absolute Difference (MAD) Matching Criterion:

The number of arithmetic operations in equation (16) can be reduced by using the mean absolute difference as the best block matching criterion. MAD between the current and reference block is defined as,

$$\text{MAD}\ (d_{x,}d_y) = \frac{1}{M_1 N_1}\sum_{(m,n)\in W}|b[m,n,k] - b[m-dy, n-dx, k-1]| \quad (2.17)$$

MAD is also known as city block distance. It is a very popular technique especially suitable for VLSIimplementation and is also used in Moving Picture Expert Group (MPEG) standard.

2.1.3.3 Sum of Absolute Difference (SAD) Matching Criterion:

Sum of absolute differences (SAD) is an algorithm for measuring the similarity between image blocks. It works by taking the absolute difference between each pixel in the original block and the corresponding pixel in the block being used for comparison or reference block.

Thus, SAD is defined as

$$\text{SAD}\ (d_{x,}d_y) = \sum_{(m,n)\in W}|b[m,n,k] - b[m-dy, n-dx, k-1]| \quad (2.18)$$

2.1.3.4 Matching Pixel Count Matching Criterion:

In matching pixel count (MPC) criterion, every pixel in the rectangular block within the search window is divided as matching and mismatching pixel according to below condition,

$$C(m,n;d_x,d_y) = \begin{cases} 1 & if \quad |b[m,n,k] - b[m-dy, n-dx, k-1]| \le T \\ 0 & Otherwise \end{cases} \quad (2.19)$$

Where T is a predetermined threshold. Then the matching pixel count is the count of matching pixels and is given by

$$\text{MPC}\ (d_{x,}d_y) = \sum_{(m,n)\in W} C(m,n;d_x,d_y) \quad (2.20)$$

In order to reduce the computational complexity, fast searching algorithms nave been developed. Various methods are proposed, some of them have been used in the JM software (JVT reference software of H.264/AVC).

2.2 Types of fast BMA

Fast block matching algorithm can be categorized into six categories [3]:

1. Fixed set of search pattern
2. Predictive search
3. Hierarchical or Multiresolution search
4. Sub-sampled pixel or Matching-error computation
5. Bitwidth Reduction
6. Fast full search

2.2.1 Fixed set of search pattern:

In this category of search pattern a subset of possible search candidate locations are selected. Many search algorithm have such pattern like Full search (FS)[6], Three step search(TSS)[8], four step search (4SS)[11], New three step search (NTSS)[8], Diamond search (DS)[12], Cross diamond search(CDS)[14], Simple and efficient search algorithm (SESTSS)[9],etc.

2.2.2 Predictive Search:

In the predictive search pattern, the correlation between the current block and its neighboring block is used in the spatial or temporal domain to predict the target motion vector and the initial motion is estimated. The predicted motion vector can be determined by selecting one of the neighboring motion vector according to certain criteria or by calculating the statistical average of the neighboring motion vectors, for example the predictors are obtained from the motion vectors of the macroblock on the left, top, top right, and their median, also the predictors can be the motion vector of the collocated macroblock in the previous frame, or in the previous two frame[3]. The predictive motion estimation can significantly reduce the search area as well as the computation. Note that additional memory for storing the neighboring motion vectors is required in this method. This technique is generally used in adaptive rood pattern search (ARPS)[15],unsymmetrical multi-hexagon search (UMHexagonS) [18], and simplified block matching algorithm for fast motion estimation [9].

2.2.3 Hierarchical or Multiresolution Search:

Methods in this category generally explore the correlation between different resolution levels of representation of the same frame .In hierarchical search methods; the frame size at different resolution levels is identical to each other, while the size of the macroblock varies. Thelower level having larger macroblock and the higher level having smaller macroblock. It is assumed that larger macroblock's motion vector in the lower level can be used as a good prediction of the

motion vector for the macroblock those are in the higher level. However this assumption often mismatches the actual situation, and consequently could lead to poor performance. Another approach to reducing the number of searches is to estimate the amount of motion progressively from lower resolution to higher resolution images. The temporally adjacent images are first decomposed into a number of levels, with each higher level having a lower resolution. This is known as multiresolution representation also called pyramidal structure.The idea behind the multiresolutionschemeis based on predicting an initial estimate at the coarse level and refining the estimate at the fine level. Usually two or three level hierarchical search is adopted in this technique. The search range at the fine level is much smaller than the original search range. More levels can save more computation, but the probability of being trapped in local minimum is higher because when the image is scaled down, the detailed textures will be lost. In fact, the multiresolution technique is used as one of the most efficient methods in BMA and is mostly used in images with very large frames and search areas [3].

2.2.4 Sub-sampled pixel or Matching-error computations:

The previous three patterns of block matching algorithms reduce the computation of motion estimation by reducing the number of search locations. This technique is based on reducing the number of pixels that have been used to compute the error between the current macroblock and reference macroblock. For example the sub-sampled method may use only a fraction of pixels within a macroblock by performing 2 : 1 pixel sub-sampling in both horizontal and vertical directions. So the result of the total computation can be reduced by a factor of 4. Such method of reducing computation can be incorporated in other block matching algorithms to achieve higher computational gain.

2.2.5 Bitwidth Reduction:

In luminance frame each pixel is represented in eight-bit resolution. This search technique is useful to reduce the bitresolution from 8 to one or two so that it can reduce the hardware cost and power consumption, and then applying conventional motion estimation search algorithms. For example the block mean is used as a threshold to represent each pixel with a single bit-plane by using one-bit transform (1BT). The bit-plane of an image frame is constructed as follows:

$$B(i,j) = \begin{cases} 1, & if\ I(i,j) \geq t_{bm} \\ 0 & Otherwise \end{cases} \quad (2.21)$$

Where t_{bm} is the threshold value, $I(i,j)$shows the (i,j)th pixel of the image frame,$B(i,j)$shows the corresponding bit-plane value.

Similarly the bit plane of the search window can be constructed.

2.2.6 Fast Full Search:

Fast full search pattern tries to improve the time toconstruct the matching block without compromising the quality of the full search. However, researches invented that the quality of the produced compressed videos is not as good as the one produced by full search. Fast full search techniques initially make a simple check to determine whether a candidate block is possible to be the matchingone or not. Then, only the potential candidate blocks are furtherprocessed for detailed distortion calculation. Thus, a large numberof unnecessary computations for impossible candidate blocks can be avoided. For example, successive elimination algorithm (SEA) eliminates unnecessary candidate blocks by checking ifthe absolute difference between the summation of current block pixels and the summation of candidate block pixels is larger than the up-to-date minimum SAD(sum of absolute difference), denoted as SADmin, then this candidate block should be skipped. If the difference is smaller than SADmin then, SAD calculation between these two blocks is still necessary and the new SAD becomes SADmin. Therefore, the computational complexity of the checking procedure is very small, and skipping of unnecessary candidate blocks can speed up the full searchprocess. Note that the skipping ratio will be increased if the initial SADmin is smaller than others [6].

2.3 Search Techniques:

After choosing a suitable criterion, the next step is to estimate the motion vector for the block within the search window which satisfies the chosen criteria. Few of them are discussed below,

2.3.1 Exhaustive or Full Search:

To find best matching moving block using any of the criteria which are described above, we need to compute the appropriate matching metric within the search window for all possible integer-pel displacements. This results in the exhaustive search or also called as full search [5]. Generally full search is the optimal search method but it's computationally intensive. Suppose M_1xN_1 is the block size in the current frame and M_2xN_2 is the block size in the reference frame and if M_1=N_1=M_2=N_2=8, then the number of operation per block can be obtained as follows,

Number of searches/block = $2M_2 \times 2N_2 = 256$ (2.22)

Number of operations/search = $M_1 \times N_1$ (2.23)

For minimum MSE criterion,

$$1 \text{ operation} = (M_1 - 1) \times (N_1 - 1) ADD + M_1 \times N_1 MUL$$

$$= 49 \text{ ADD} + 64 \text{ MUL} \quad (2.24)$$

Therefore, total number of arithmetic operations required for the full search using the minimum MSE criterion is,

Total number of operations/block = Number of search/block × Operations/search ×(ADD + MUL)/Operation

$$= 256 \times 64 \times (49\, ADD + 64\, MUL)$$

$$\approx 1{,}048{,}576\ (ADD + MUL) \quad (2.25)$$

For different format of image the number of arithmetic operations will be different.

2.3.2 Three Step Search:

This is a fine coarse search mechanism. As the name implies, the motion vector corresponding to the best matching block is obtained in three steps. Each step involving only nine calculation of the chosen metric [8]. This search is sub-optimal because it does not search exhaustively for the best matching block. Figure 2.2 shows the pattern of three step search algorithm. The steps of this algorithm are given below:

Step-1: The matching metric is computed at nine pixel location marked as '×' i.e. 9 × 9 search window, with the center point corresponding to zero motion vector. The pixel separation will be 4.

Step-2: Calculate the minimum value. Shift the center to the minimum value and again the metric is calculated at the pixel marked by '0', with a pixel separation of 2 i.e. 5 × 5 search window around the locations determined by the first step.

Step-3: Again calculate the minimum value, shift the center to the minimum value and the metric is calculated at the pixel marked by 'Δ', with pixel separation of 1 i.e. 3 × 3 search window.

The final step yields the motion vector. The three step search is one of the most popular block matching algorithm because of its simplicity and effectiveness. For maximum displacement window 7 i.e. d=7, the number of checking points required is (9+8+8)=25. For larger search window, three step search can be easily extended to n-steps using the same searching strategy with number of checking points required equal to [1+8(log2 (d+1))].

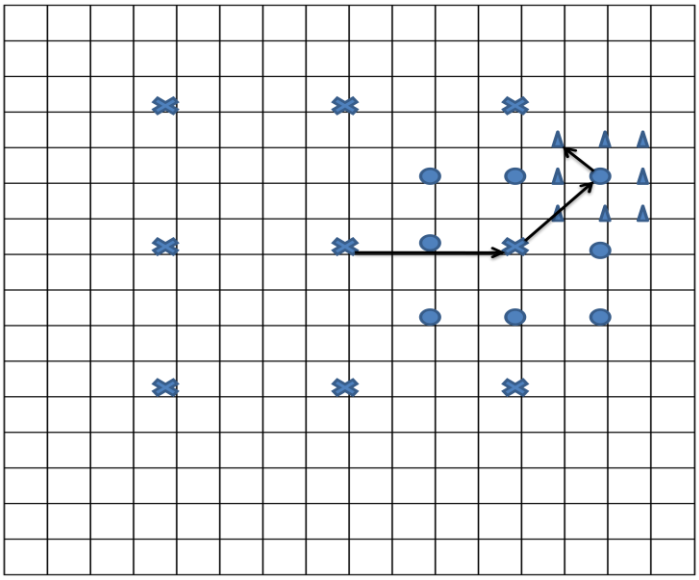


Figure 2.2: Three Step Search Procedure

2.3.3 New Three Step Search:

The result of three step searches can be improved by providing a center biased searching scheme and to reduce computational complexity it's having the provision for half way stop. It was one of the first effective and widely accepted fast algorithm and is frequently used in earlier standards like MPEG-1 and H.261. The three step search uses a fixed pattern. It uses uniformly allocated checking patterns for motion detection and it is prone to missing small motion. The process of this search technique is shown in Figure 2.3[9].

In the first step 16 points are checked in addition to the search origin for lowest weight using cost function. In the initial stage from the center of the search 8 are located at the distance of S=4 and including that 8 are located at the distance of S=1 away from the search origin. If the lowest cost is at origin then the search is stopped right there and the motion vector is set to (0, 0). If the lowest weight is any one of the 8 locations at S=1, then the center is shifted and check for the weights adjacent to it. Depending upon the place of the point again we need to check 5 points or 3 points (Figure 2.3). The location that gives the lowest weight will be the best match and the motion vector is set to that location. On the other hand, if the lowest weight after the first search was one of the 8 locations at S=4, then we follow the normal three step search procedure [8]

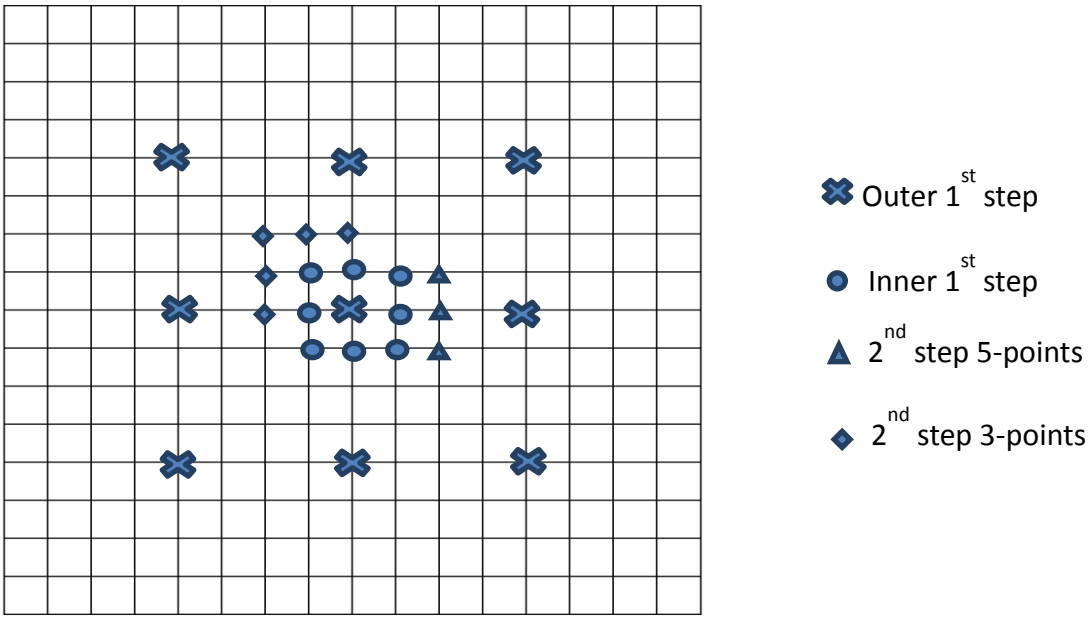


Figure 2.3: New Three Step Search procedure

2.3.4 Simple and Efficient Search (SES):

Simple and efficient search [10] is another extension to Three step search. It is based on the assumption of unimodal error surface. The basic idea behindthe algorithm is that for a unimodal surface there cannot be two minimums in opposite directions and hence the 8 pointfixed pattern search of three step search can be changed to incorporate thisand save on computations. The algorithm has also same pattern as three step search, but thechange is that each step has further two phases. The searcharea is divided into four quadrants and the algorithm checksthree locations A,B and C as shown in Figure 2.4. A is at theorigin and B and C are located at S = 4 away from A inperpendicular directions. Depending on certain weightdistribution amongst the three the second phase selects someadditional points (Figure2.5). The method for determining a search quadrant for second phase is as follows:

If MAD(A) ≥MAD(B) and MAD(A) ≥ MAD(C), select (b)

If MAD(A) ≥ MAD(B) and MAD(A) ≤ MAD(C), select (c)

If MAD(A) < MAD(B) and MAD(A) < MAD(C), select (d

If MAD(A) < MAD(B) and MAD(A) ≥ MAD(C), select (e)

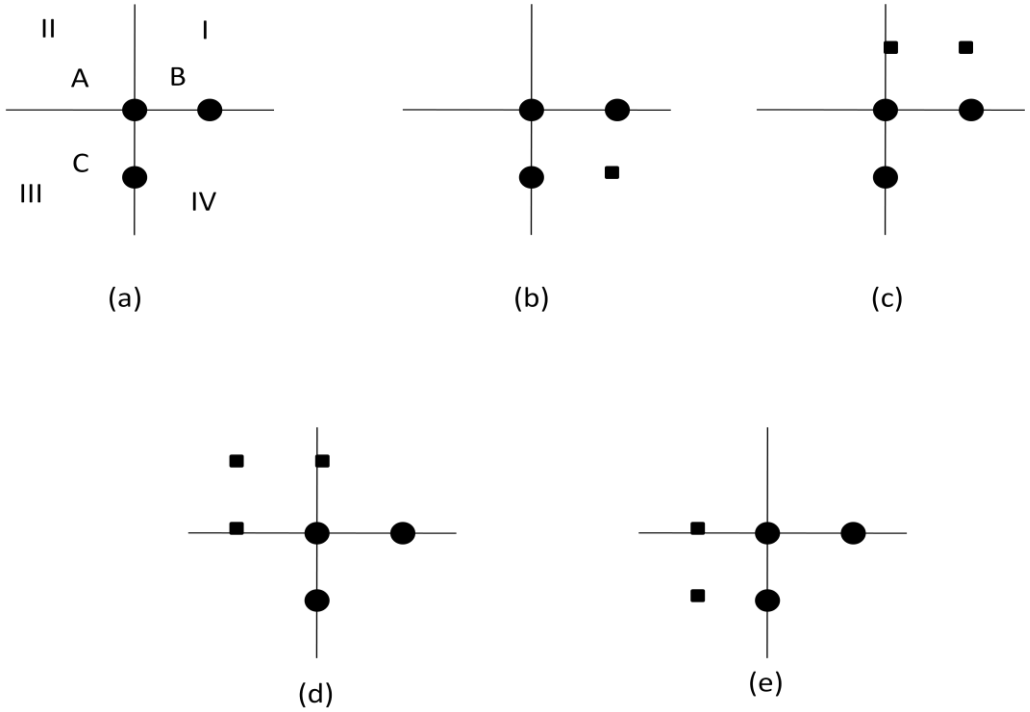


Figure 2.4: Search patterns corresponding to each selected quadrant: (a) Shows all quadrants (b) Quadrant I is selected (c) Quadrant II is selected (d) Quadrant III is selected (e) Quadrant IV is selected

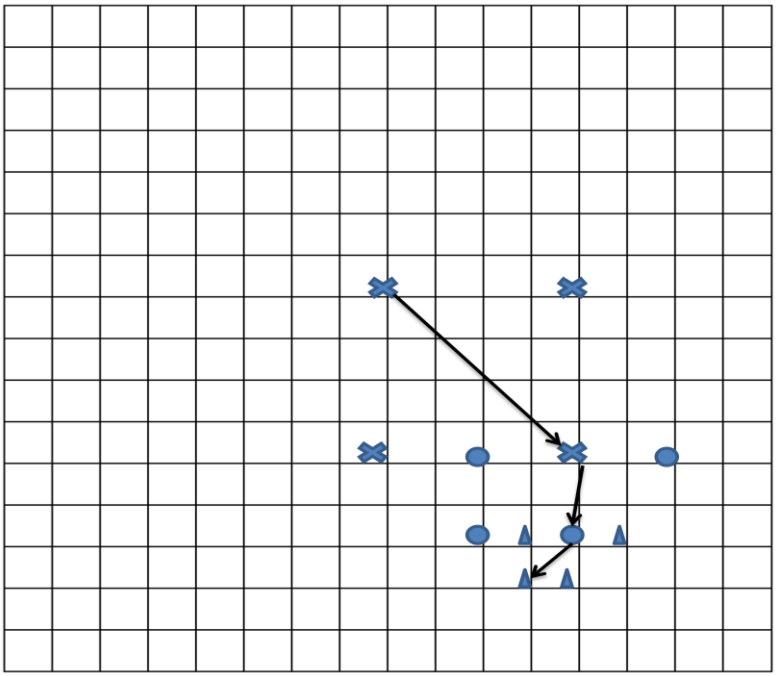


Figure 2.5:SES procedure

After getting the suitable quadrant find the location with minimum weight and set it as origin and again apply three step search until S=1. The steps of the algorithm are given below:

Step-1: Find out the 9 points as in the case of three step search at S=4 then follow the above rule and find out the suitable quadrant.

Step-2: Then find out the point having minimum weight and shift the point as origin then reduce the search area to S=2 and get 8 points and among them to find out the suitable quadrant.

Step-3: Again locate the point with minimum weight and set it as origin and at S=1 find out the 8 points and the minimum value will give the motion vector.

2.3.5 Four Step Search:

Similar to NTSS (New Three Step Search) [9], 4SS (Four Step Search)[12] also employs center biased searching and has a provision of halfway stop. Usually Four Step Search employs a fixedpattern of size S = 2 for the first step, for any value of thesearch parameter. Hence it checks at 9 locations in a5x5 window. If the least weight is found at the center of searchwindow the search jumps to fourth step. If the least weight is atone of the eight locations except the center, then we make it as the search origin and further move to the second step. The search window is still constant at 5x5 pixels. The procedure can end up at 3 locations or 5 locations depending upon the position of the least weight. The patterns are shown inFigure 2.6. Once again if the least weight location is foundat the center ofthe 5x5 search window then we jump to fourth step or else further move on to third step. The third step is very similar to the secondstep. In the fourth step the window size is reduced to 3x3, i.e.at S = 1. The location with the least weight is the best matchingmacro block and the motion vector is set to point of thatlocation. Figure 2.7 shows the pattern of four step search.

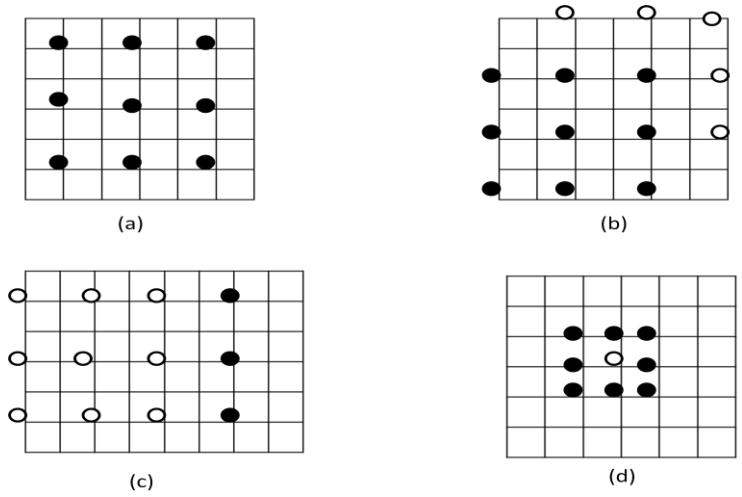


Figure 2.6: Search pattern of the FSS

(a) First step (b) Second/third step (c) Second/third step (d) Fourth step

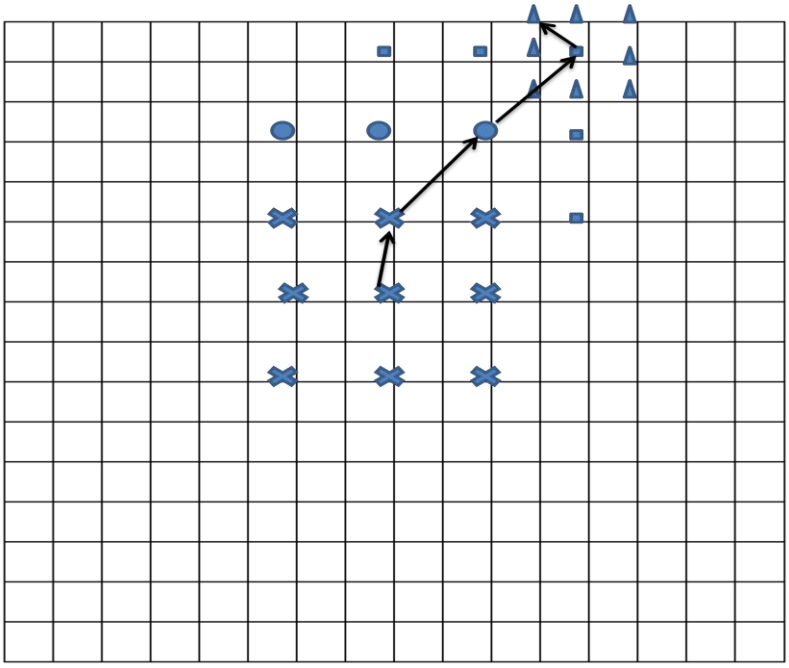


Figure 2.7: Four step search procedure

In summary,

Step-1: At S=2, find out the 9 points and calculate the least weight if it is at the center position then go to step-4.

Step-2: If the least weight is at corner place as shown in Fig.2.6 (b) then find out additional five points and calculate least weight.

Step-3: If the least weight is at side position as shown in Fig.2.6 (c) then find out additional three points and determine the least weight position.

Step-4: At the least weight position again search 8 points at S=1 and among them the least weight position will give the motion vector.

2.3.6 Diamond Search (DS):

DS [12] algorithm is very much similar to 4SS(Four step search), but the search point pattern is changed from a square pattern to a diamond shape pattern, and thereis no limit on the number of steps.Diamond search uses two different types of fixed patterns, one is Large Diamond Search Pattern (LDSP) and the other is SmallDiamond Search Pattern (SDSP). These two patterns and theDS procedure are illustrated in Figure 2.8. As in FSS, the initialstep uses LDSP and if the least weight is at the center locationthen jump to fourth step. All the consecutive steps, except the laststep, are similar and use LDSP, but the number of pointswhere cost function is checked are either 3 or 5 it depends upon the position of the least weight point. The procedure is shown inFigure 2.9. The last step uses SDSP around the shifted origin having least weightand found the best match. The benefit of using this algorithm is, it can find the global minimum very accurately

because thesearch pattern is neither too small nor too big and there is no limit to the number of steps.

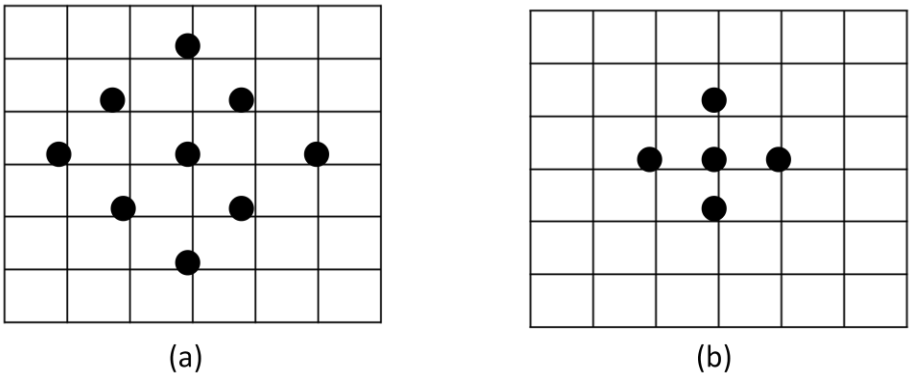


Figure 2.8: Two different pattern of DS (a) LDSP (b) SDSP

In summary,

Step-1: Using LDSP calculate 9 points and using cost function calculate the minimum value if it at the center position then go to step-3.

Step-2: If the minimum value is at any position around the center then search additional 3 points or 5 point according to the position. Use any cost function to determine the least weight. Repeat the step until the minimum value is at the center position.

Step-3: Use SDSP at the center and determine the least value using cost function and obtain the best match.

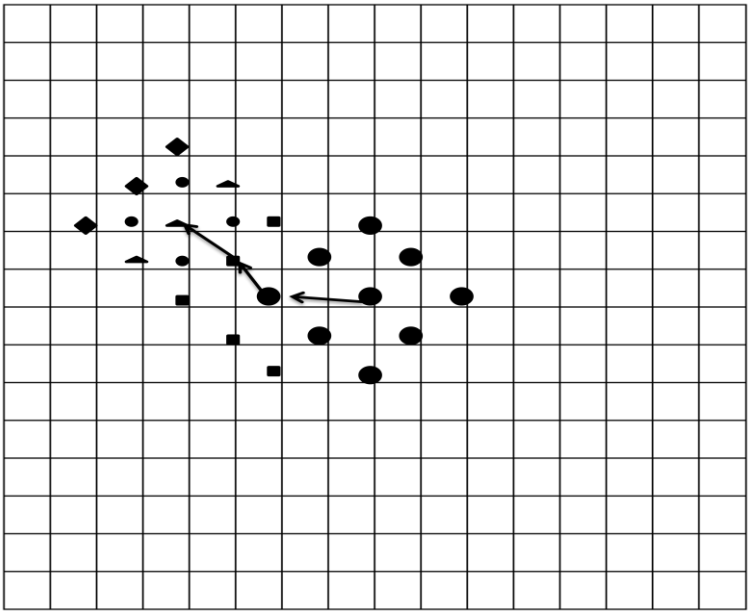


Figure 2.9: Diamond Search procedure

2.3.7 Adaptive Rood Pattern Search:

ARPS [16] algorithm is based on the concept that normally the motion in a frame is coherent, that means if the macro blocksaround the current macro block shifted in a particular directionthen there will be a high probability that the current macro blockwill also have a similar motion vector. This algorithm employs themotion vector of the macro block to its immediate left topredict its own motion vector. An example is shown in Figure 2.10.This pattern also checks at a rood pattern distributed points, as shown in Fig.2.10, at a step size of S = Max (|M|, |N|). M and N are the x-coordinate and y-coordinate of the predictedmotion vector. This rood pattern search is always the initial step. It directly implements the search pattern in an area where there is a highprobability of finding a good matching block. The point thathas the least weight becomes the origin of the search pattern for further searchsteps, and the search pattern is changed to SDSP. The procedure is repeated in SDSP until least weighted point is found to be at the origin of the SDSP. This algorithm is further improved by adding Zero MotionPrejudgment [8], using which the search will be stopped half way if the least weighted point is already at the center of the rood pattern. The measure benefit of this algorithm over diamond search is if the predicted motion vector is (0, 0), it does not waste time in doing LDSP, rather it directly startsusing SDSP, and if the predicted motion vector is faraway from the center, then again adaptive rood pattern search algorithm save number of computationsby directly jumping to that point and using SDSP, whereasDS waist its time in doing LDSP. We need to take care for not to repeat the calculation for same point and if the predicted motion vector is matched to the rood pattern we need to avoid double calculation. For macro blocks inthe first column of the frame, rood pattern step size is fixed at2 pixels.

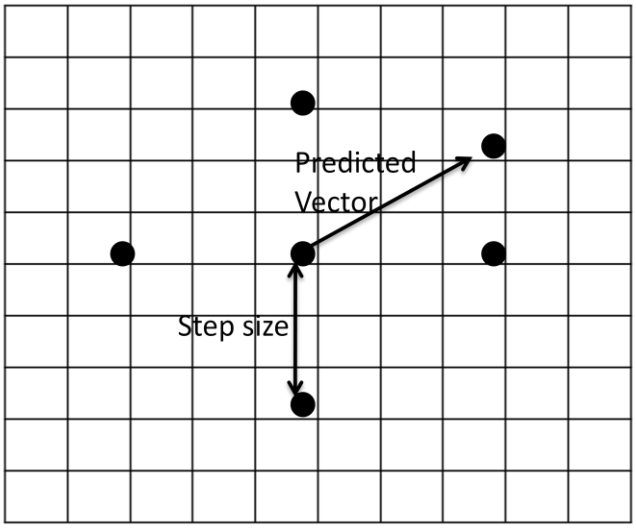


Figure 2.10: Adaptive Rood Pattern: The predicted MV is (3,-2) and the step size S=Max (|3|, |−2|) =3

2.4 Performance Study:

For the above search techniques the performance is measured in no. of search point per macroblock and the performance of PSNR with respect to frame number. The result is generated in frame-by-frame analysis. Three standard sequences are taken to study the discussed algorithm i.e. Exhaustive or Full Search (ES),Three Step search (TSS), New Three Step Search (NTSS), Simple and Efficient Search (SES), Four Step Search (SS4), Diamond Search (DS), Adaptive Rood Pattern Search (ARPS). The standard sequences are (1) Caltrain, (2) Missa and (3) Salesman.

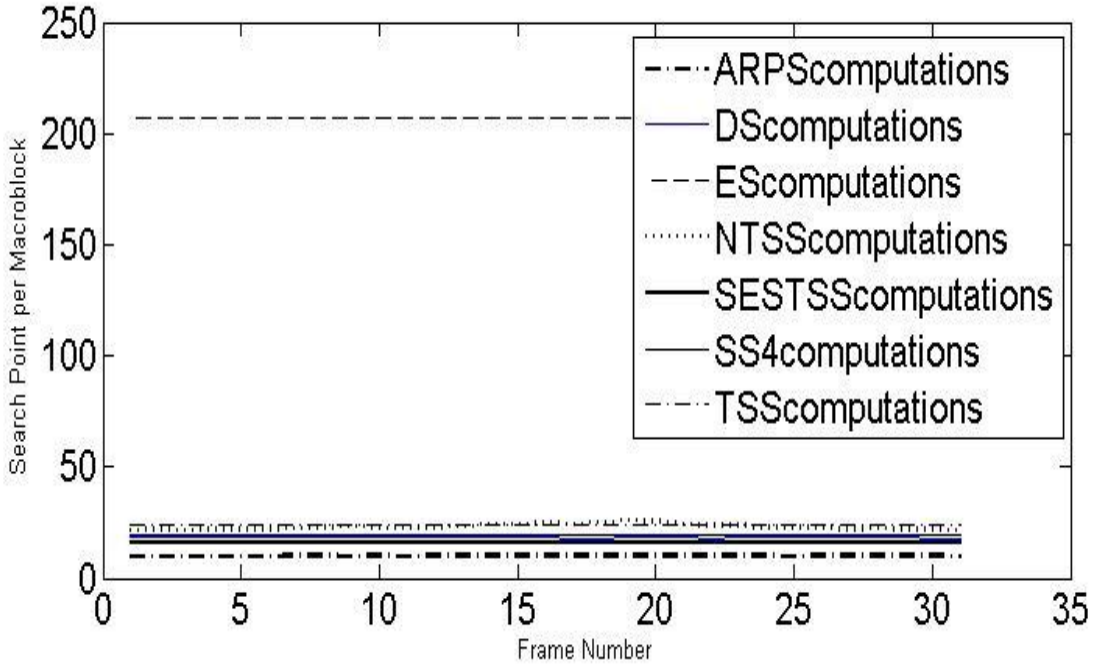


Figure 2.11: Performance of search point at different frame no. for Caltrain Sequence

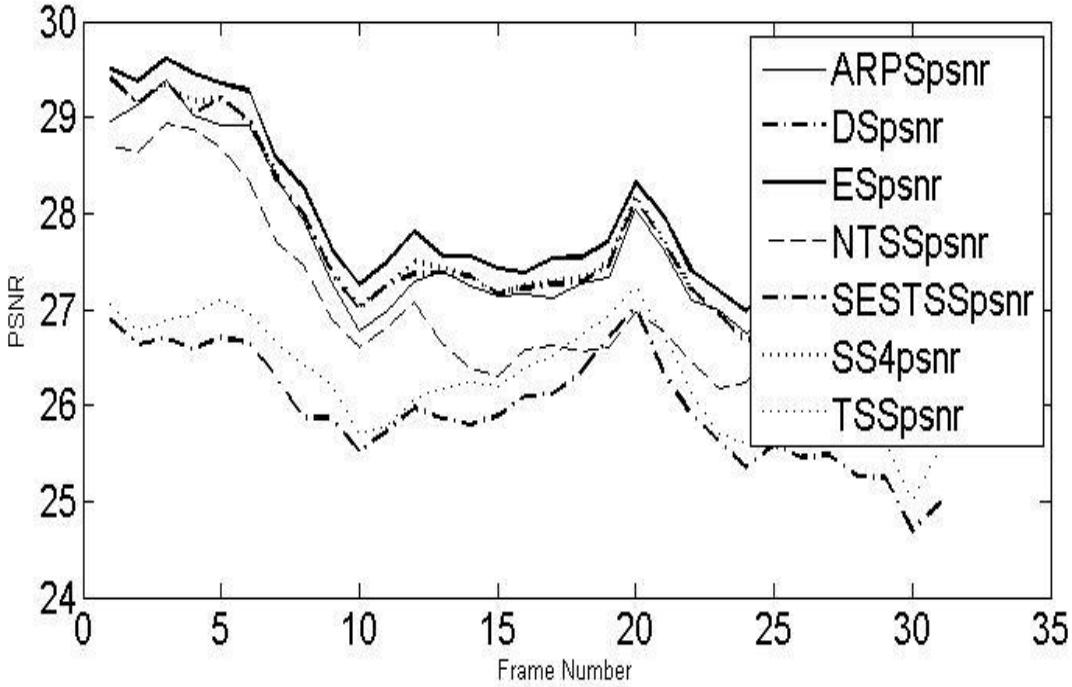


Figure 2.12: Performance of PSNR at different frame no. for Caltrain Sequence

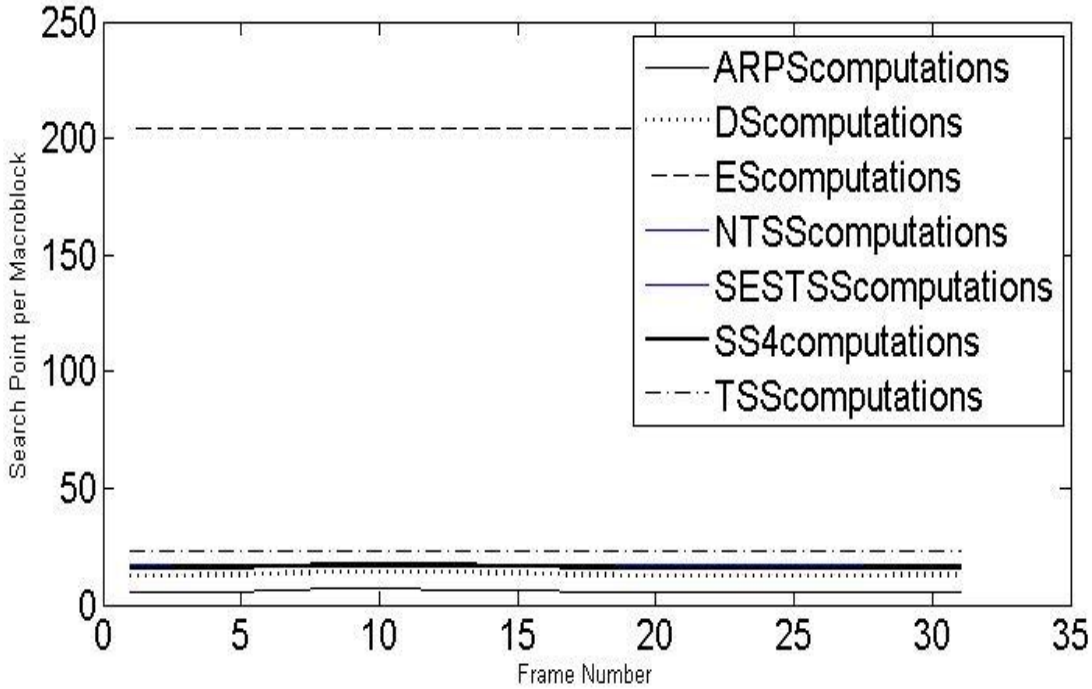


Figure 2.13: Performance of search point at different frame no. for Salesman Sequence

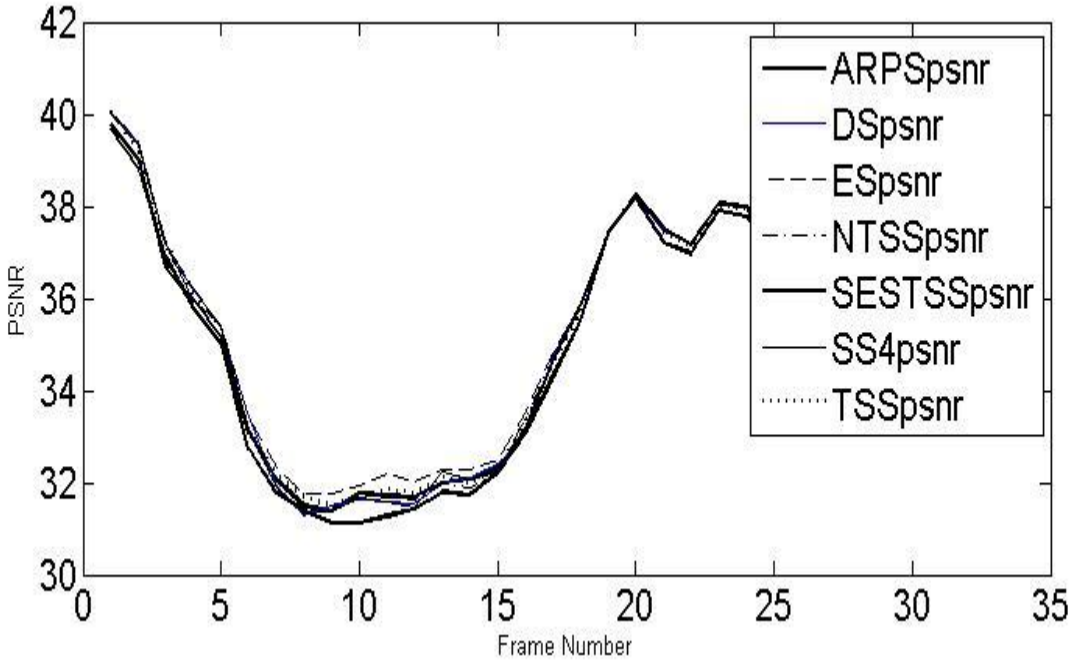


Figure 2.14: Performance of PSNR at different frame no. for Salesman Sequence

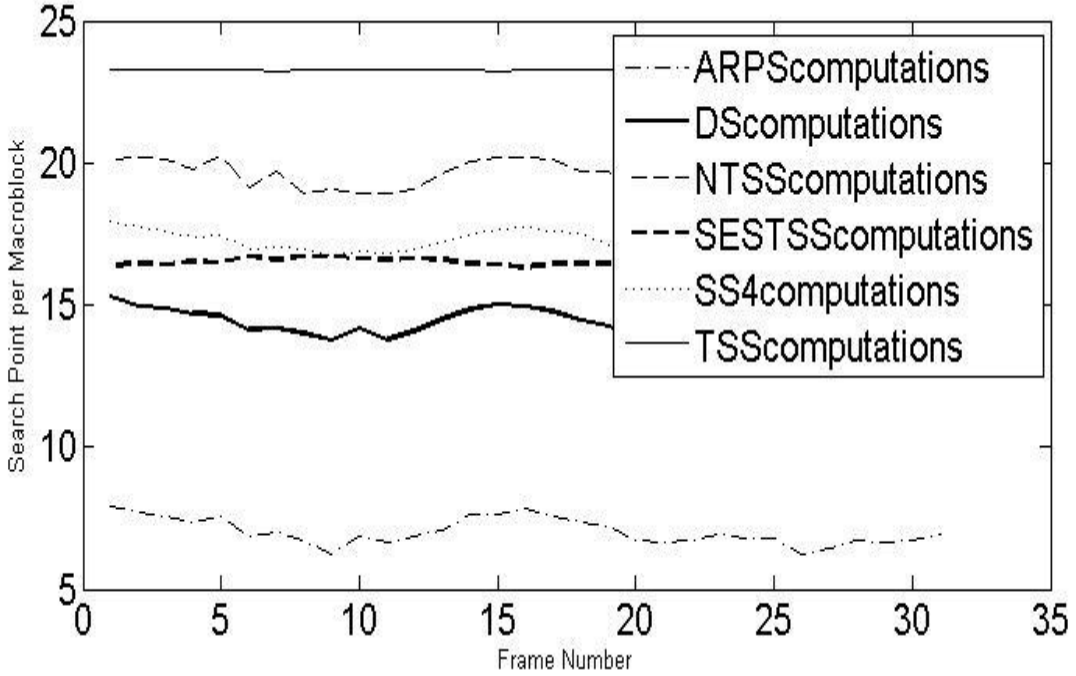


Figure 2.15: Performance of search point at different frame no. for Missa Sequence

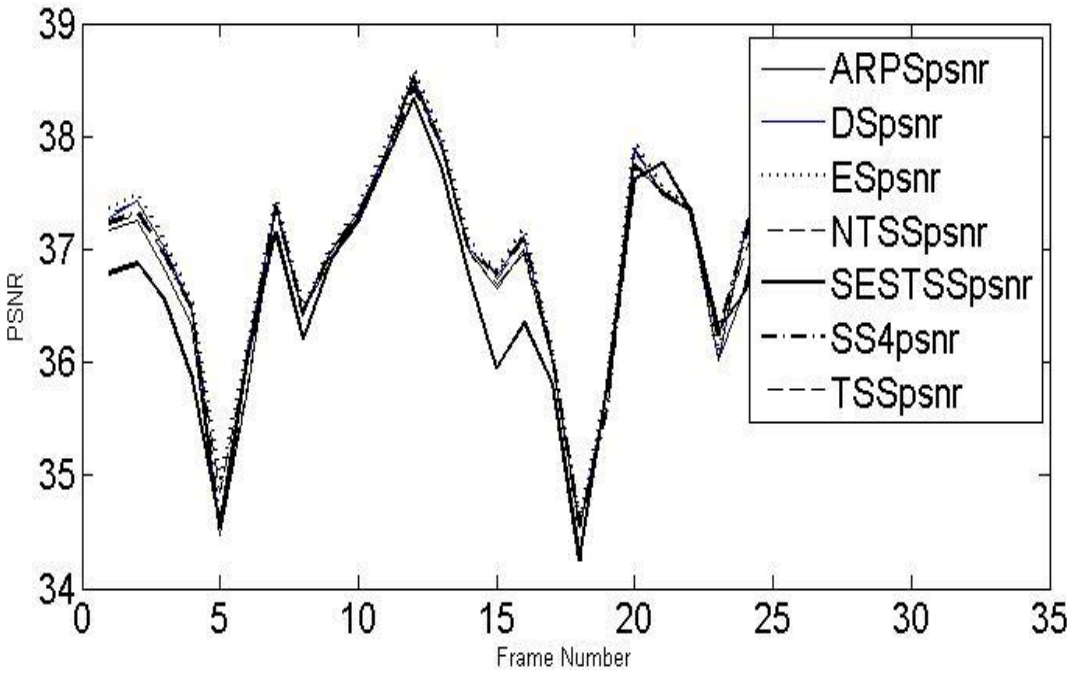


Figure 2.16: Performance of PSNR at different frame no. for Missa Sequence

2.5. Observations:

From the above plots it is clear that the average no. of search point is less in the adaptive rood pattern search (ARPS) after that the diamond search is coming into account while it is higher in case of full search (FS) in almost every sequence. The simple and efficient search (SESTSS), three step search (TSS) and four step search (SS4) has almost similar search point. The PSNR performance of ARPS is worst among all which in turn gives better quality. So in these techniques the ARPS is better in comparison to others. The PSNR performances of all algorithms are clearly visible in Caltrain sequence. In other two sequences the performance is not that much clear.

CHAPTER-3

Early Termination using Adaptive Threshold

3.1 Introduction:

Motion estimation (ME) has been adopted by all of the existing standards related to video coding such as the MPEG series [3] and H.26x series [1] to remove temporal redundancy between successive frames. Motion estimation is the most computational intensive part, consumes up to 50% of video encoding time [3]. The Full Search (FS) [6] block matching algorithm is the best in terms of quality and simplicity, but it is the most computation intensive algorithm. Various fast block matching algorithms has been proposed such as three-step search (3SS) [8], new-three-step search (NTSS) [9], four-step search (4SS) [12], and diamond search (DS) [13], have been proposed to reduce the computational complexity of ME module.

The main objective of early termination process is to reduce computation with minimum effect on video quality. Generally at each step in the search algorithm, the threshold is checked, if the search point is less than or equal to a predefined threshold. Otherwise the algorithm proceeds to the next step in the algorithm. So the best search point for the current block can be found early. An adaptive threshold has been introduced in [20];

3.2 Adaptive threshold selection:

The thresholds are determined by two factors: the average BDM (block distortion motion) information from the coded blocks of the previous frame and the empirical factor values [20]. The definition is given as:

$$Threshold(f_i) = BDM_{avg}(f_i - 1) * \beta_i \qquad (3.1)$$

$$i = 1,2,3 \qquad \beta \in 0 \sim 1$$

$BDM_{avg}(f_i - 1)$ is the average block distortion motion for each search point on the previous frame $f_i - 1$. β_i is used to restrict the threshold value of the threshold (f_i) at each step to avoid being trapped in the local minimum due to early termination. The block distortion motion is generally calculated using any of the cost function method.

Here we used minimum absolute difference (MAD) to evaluate cost function. In this method the search point with minimum BDM is calculated at each step. After acquiring the minimum BDM in first step, it still needs to compare with the minimum BDM of the previous step to decide the smallest BDM. This is because the minimum BDM of the previous step may still be the smallest

BDM. In addition to that, it can still compare with the threshold value of the current step to decide the early termination operation.

3.3 Proposed Modified Three Step Search (TSS):

The general idea is already discussed in chapter 2. It starts with the search location at the center and sets the 'step size' S = 4, then it searches at eight locations ± S pixels around location (0,0). From these nine locations, by using cost function it picks the one giving least cost and makes it the new search center. Then it sets the new step size S = S/2, and repeats similar search for two more iterations until S = 1. At that point it finds the location with the least cost function and the macro block at that location is the best match. The calculated motion vector is then saved for transmission.

The disadvantage of this method is that even if the minimum the motion vector is found in early stages it will still continue to search until the step size will reduced to one, which increases the computational complexity. So inorder to reduce the complexity the early termination method is used. Some of the fixed early termination based methods are already proposed. The method of proposed modified TSS is described in Figure 3.1.

The proposed modified three step search works as follows: after evaluating 9 cross point in the first step, find the minimum BDM using MAD (minimum absolute difference), then check whether the point is less than or equal to the threshold, if yes stop the search algorithm and take this point as the best MV for this block, if not then reduce the step size to two then check for eight points around the minimum value and calculate the minimum value then check whether the minimum value is less or equal to the threshold, if yes then stop the search algorithm and obtain the MV, if not then reduce the step size to one and find eight points around the minimum value then check for the minimum value among those eight points, the minimum value will give the best matching block and the location will give the MV.

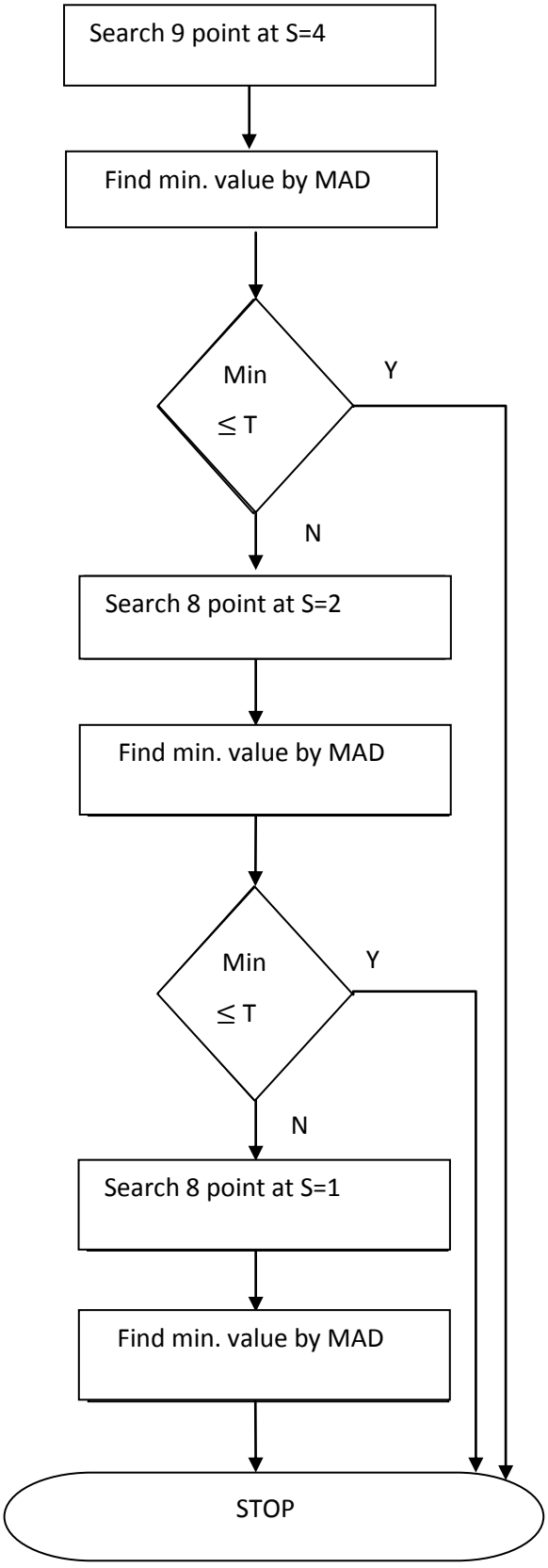


Figure 3.1: Flow chart for Modified TSS

3.4 Proposed Modified Four step Search (4SS):

The four step search algorithm has four steps: step1 sets a fixed pattern size of S = 2 and search for 9 locations in a 5x5 window. If the least weight is found at the center of search window the search jumps directly to fourth step. Otherwise, step 2 take the point having the least weight as center point but the search window is still maintained as 5x5 pixels wide as already discussed in chapter 2. Step 3 is exactly the same as the second step. In the fourth step the window size is reduced to 3x3, (i.e. S = 1) the location with the least weight is the best matching MV.

The disadvantage of four step search algorithm is, the search will jump to the fourth step of algorithm only when the minimum value is at the center place otherwise it will search for 8 more points around that. So it is moving to the next step if the minimum value is not at center which increases the no. of search point and also increases computational complexity. To reduce the search point we have applied the adaptive threshold to the four step search. The method is described in Figure 3.2.

The threshold is applied after each step of evaluating the cost function. The proposed modified method of four step search works as follows: after finding 9 points at step size of two, the minimum value is evaluated using MAD, then the minimum value is compared with the threshold whether the value is less than the threshold or not, if yes the search will jump to the fourth step where it will find 8 point around the minimum value and get its MV, if no then again it check for the 8 point around the minimum value at the step size of two and the similar process will continue for the third step.

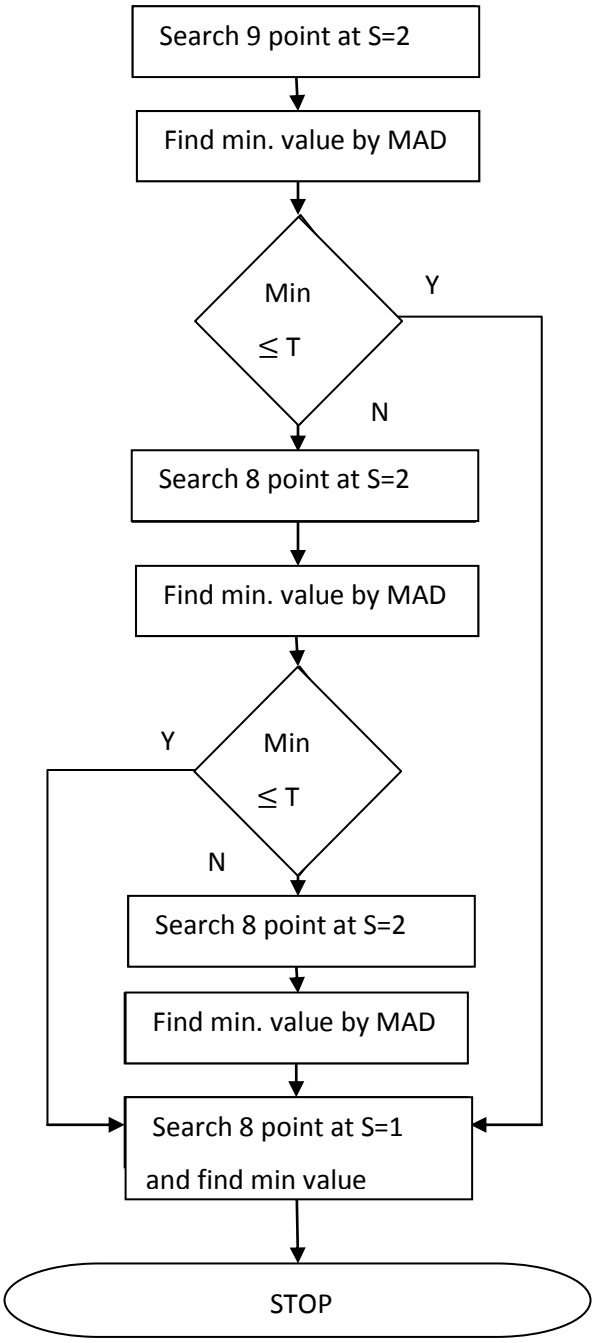


Figure 3.2: Flowchart for Modified 4SS

3.5 Results and Analysis:

To evaluate the performance of three step search using threshold and four step search using threshold we have taken four sequence named as (1) Akiyo, (2) Caltrain, (3) Foreman, (4) Table tennis as shown in Figure 3.9.Two important measures are considered for analysis, (1) Average number of search points per block, in which ithas a significant effect on motion estimation time. (2) Video quality will be evaluated by MSE. The variations of average number of search points with frame number are shown in the Figure 3.3, Figure 3.4 and Figure 3.5 taking three sequences (1) Caltrain,(2) Salesman,(3) Missa.

Sequences	BMA	Avg no. of Search Point	Motion Estimation Time(Sec.)
Caltrain	FS	202.04	120.39
	TSS	23.66	33.82
	NTSS	20.27	32.27
	4SS	24.76	44.73
	TSSTH	21.13	14.07
	4SSTH	17.37	10.09
Akiyo	FS	209.21	383.90
	TSS	24.03	101.19
	NTSS	21.57	39.14
	4SS	25.24	44.92
	TSSTH	23.09	42.75
	4SSTH	17.43	31.05
Foreman	FS	208.01	379.55
	TSS	23.66	101.69
	NTSS	20.27	40.08
	4SS	24.76	45.53
	TSSTH	21.13	42.91
	4SSTH	17.37	33.25
Tabletennis	FS	208.01	370.67
	TSS	24.02	78.56
	NTSS	22.24	37.60
	4SS	25.75	45.89
	TSSTH	23.12	42.06
	4SSTH	16.50	29.50

Table 3.1: Evaluation of Avg. no. search point and ME time

Two important measures are considered for analysis, (1) Average number of search points per block, in which ithas a significant effect on motion estimation time. (2) Video quality will be evaluated by MSE. The variations of average number of search points with frame number are shown in the Figure 3.3, Figure 3.4 and Figure 3.5 taking three sequences (1) Caltrain,(2) Salesman,(3) Missa

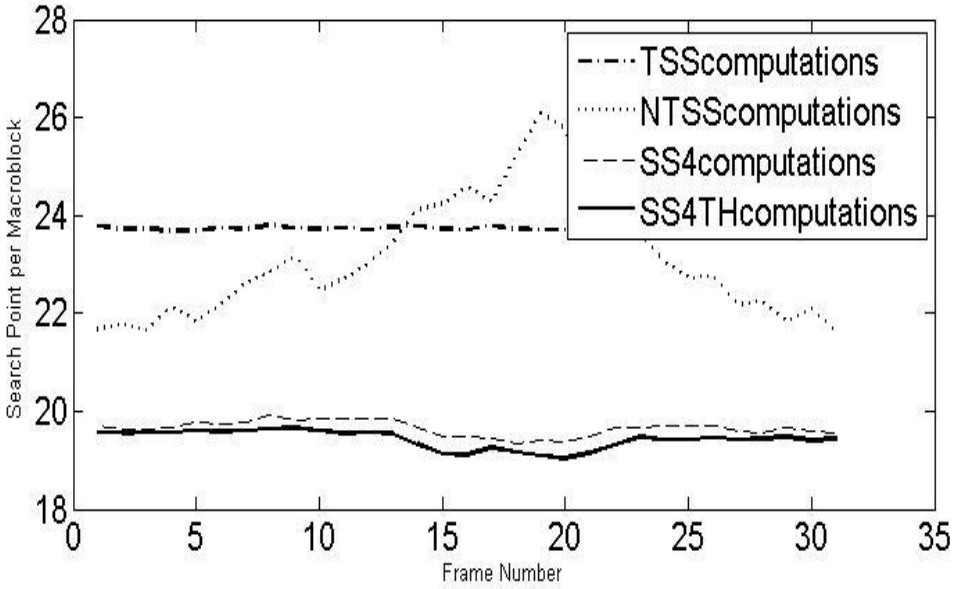


Figure 3.3: Performance of search point at different frame no. for Caltrain Sequence

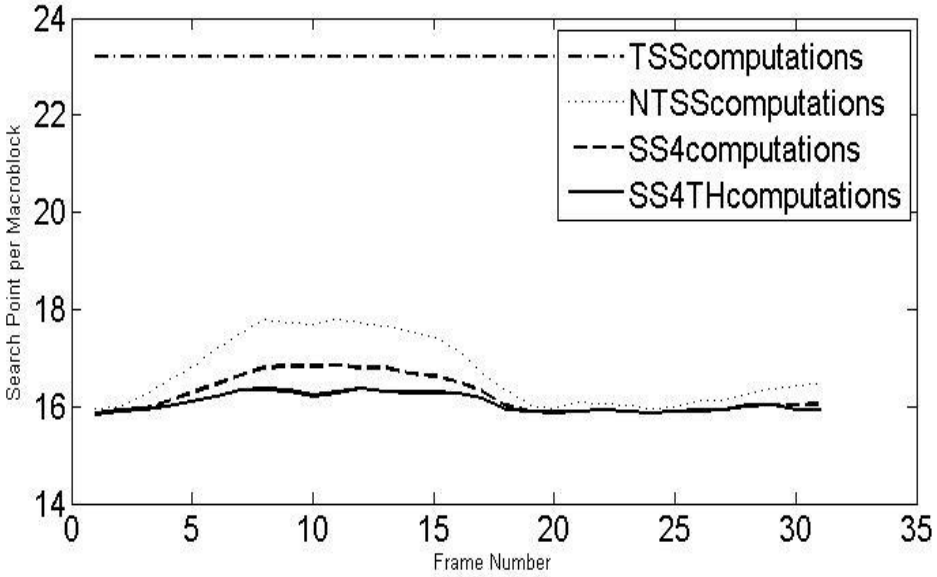


Figure 3.4:Performance of search point at different frame no. for Salesman sequence

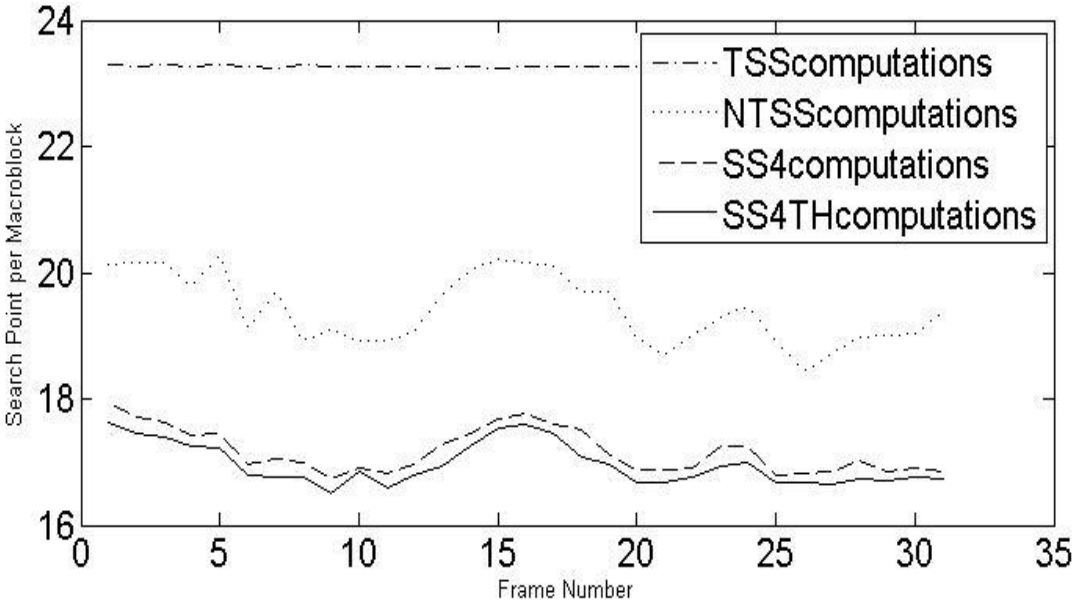


Figure 3.5: Performance of search point at different frame no. forMissa Sequence

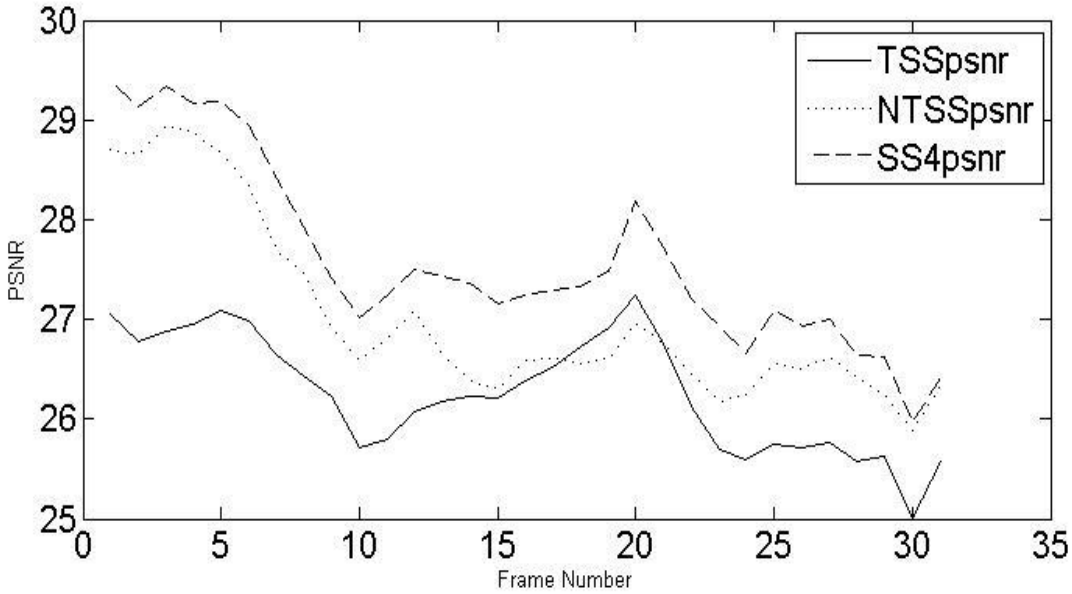


Figure 3.6: Performance of PSNR at different frame no. for Caltrain sequence

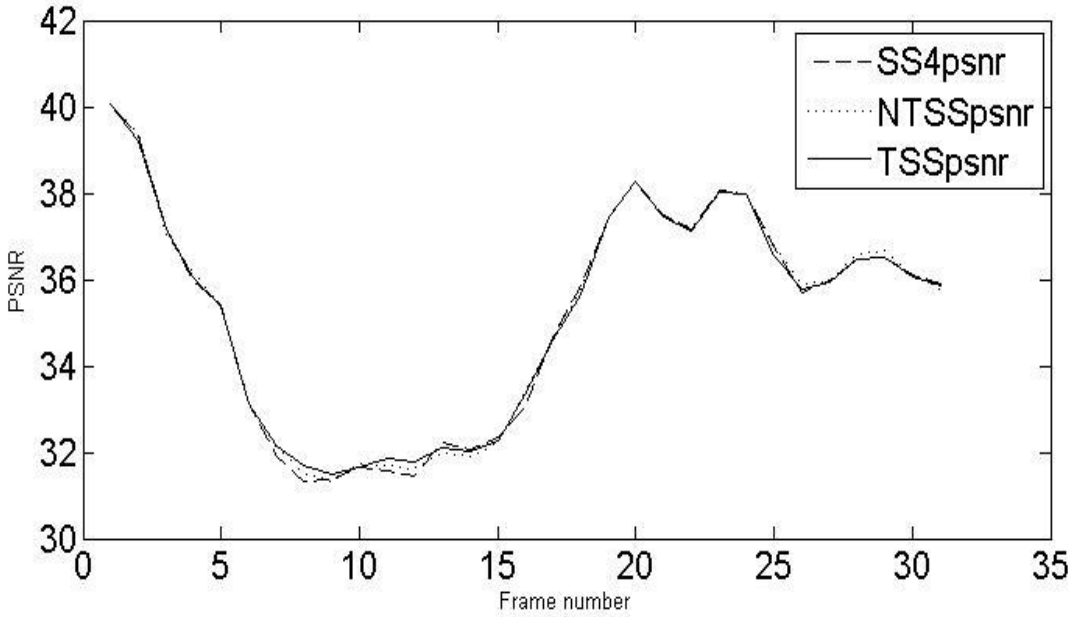


Figure 3.7: Performance of PSNR at different frame no. for Salesman Sequence

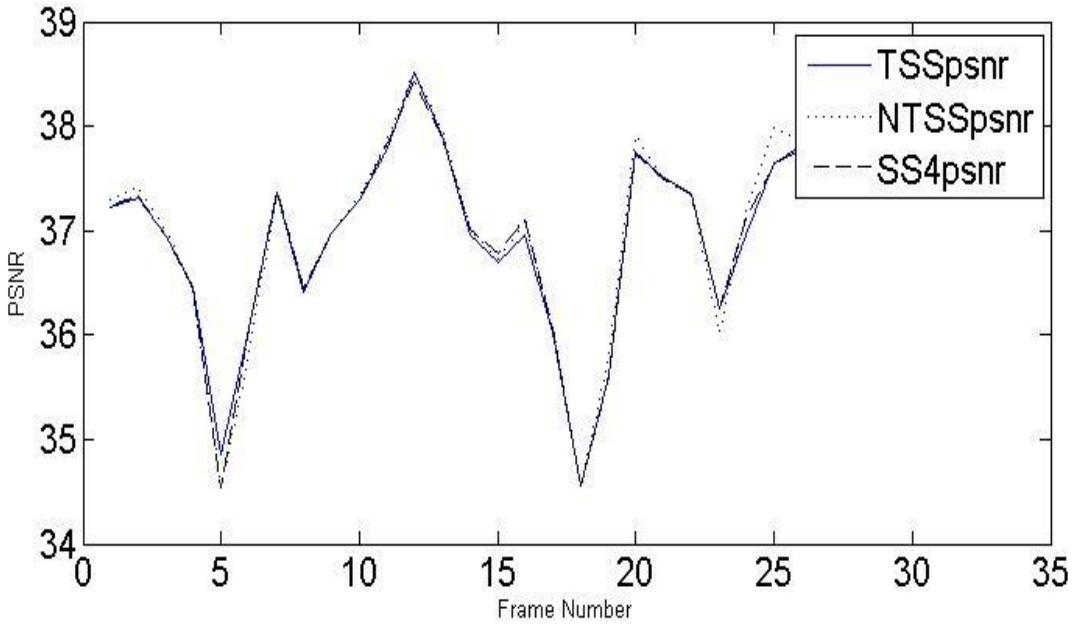
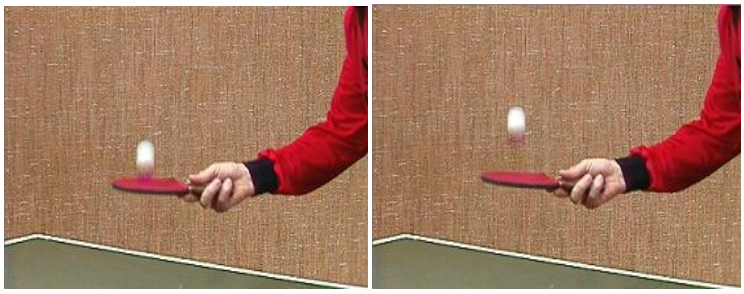


Figure 3.8: Performance of PSNR at different frame no. for Missa sequence

(a)Two frames of Akiyo Sequences

(b)Two frames of Caltrain Sequences

(c)Two frames of Foreman Sequence



(d)Two frames of Table tennis

Figure 3.9: Sequences used in algorithms

In the above table we have calculated the average no. of search point per macroblock for some of the block matching algorithm like full search (FS), three step search (TSS), new three step search (NTSS), four step search (4SS) and the proposed three step search using threshold and four step search using threshold.

From the Table-3.1, we found that the average no. of search point is quite less in case of TSSTH which is not making much difference between TSS and improved TSS but in 4SSTH it is much smaller than the other block matching algorithm. At the same time the motion estimation time is also calculated and it shows that 4SSTH is taking less time than TSSTH and other block matching algorithm. From the Figure 3.3, Figure 3.4, Figure 3.5 it is observed that improved 4SS has less search point as compared to three step search (TSS), new three step search (NTSS) and four step search (4SS).

Mean Square Error (MSE) in %				
	Akiyo Sequence	Caltrain Sequence	Foreman Sequence	Table tennis Sequence
TSS	0.16	0.12	o.16	0.11
TSSTH	0.16	0.12	o.16	0.12
4SS	0.1	0.3	0.11	0.1
4SSTH	0.1	0.31	0.12	0.1

Table 3.2: MSE values in % for conventional TSS, 4SS and Modified TSS, 4SS

In Table-3.2 the performance of quality is evaluated by measuring the value of mean square error in percentage between the existing TSS, 4SS and proposed algorithms.As it is well known that the quality of the full search is better than other block matching algorithms so we have compared the above algorithms with respect to full search and calculated the MSE. It is clear from the table that the quality of TSSTH is almost same as TSS and the quality of 4SSTH is also almost same as 4SS.So it is very difficult to show the difference in the PSNR plot which are shown in the Figure 3.6, Figure 3.7 and Figure 3.8.The PSNR performance is measured in three sequences, (1) Caltrain, (2) Salesman and (3) Missa.As the MSE value is almost same in TSS and TSSTH, the PSNR performance of TSSTH is not shown in the plots and also for 4SSTH it is not shown in the plots as the MSE value is same in 4SS and 4SSTH.So from the PSNR plots it is concluded that the quality of improved TSS and improved 4SS remains same as TSS and 4SS.So the early

termination using threshold greatly improves the four step search by reducing the no. of search point and also by reducing the Motion estimation time.

3.6 Conclusion:

In this chapter a new fast motion estimation algorithm based on H.264 has been proposed. The proposed algorithm is aimed at further reducing the searching points to accelerate the search speed without loss of its accuracy. This algorithm outperforms three step search (TSS) and four step search (4SS). The proposed algorithm carried out on MATLAB 2007. The Table 3.1 is constructed to evaluate the number of search point per macroblock as well as motion estimation time in second. Four standard sequences are taken (1) Akiyo (2) Caltrain (3) Foreman and (4) Table tennis . Here for every sequence we have taken two frames one as current frame and another as reference frame. The comparison between the number of search point and frame number is plotted in Figure 3.3, Figure 3.4, Figure 3.5.To plot these curves we have taken three sequences named as (1) Caltrain (2) salesman and (3) Missa where the result is generated frame-by-frame.The size of the macroblock is taken as 16 and the search parameter is taken as 7. The PSNR performance is also plotted for the same sequences and same parameter values for the same three sequences. The results of simulations demonstrate that the proposed algorithm canreduce the no. of search points which is quite less in case of improved TSS but in case of improved 4SS it shows a difference compared to full search (FS), new three step search (NTSS), three step search (TSS) and four step search (4SS) without loss of accuracy. Thereforethe proposed algorithmis very efficient algorithm for real time video coding application.

CHAPTER-4

Sub-pixel Motion Estimation

4.1 Introduction:

Motion estimation is a measure part of various video processing tasks such as standards conversion, frame-rate up-conversion, noise reduction, image stabilization, mosaicing and artifact concealment in archived film sequences. Basically it is a critical component of video compression systems that allowsreduction in redundancy in the temporal domain. International standards for video compressions such as MPEG-1/2/4 and H.261/3/4 now a day's employs the well-established hybrid two-component architecture, which is based on the motion estimation and compensation as well as on the lossy compression of the motion-compensated prediction error. Generally the motion estimation algorithm uses block matching in the data domain in various video compression standards. Recently there has been a lot of interest in motion estimation techniques operating in the frequency domain. These techniques are commonly based on the principle of cyclic correlation and achieving great advantages in terms of computational efficiency due to the implementation of fast algorithms.

A key performance issue in motion estimation is sub-pixel accuracy. It has shown in the previous algorithms the actual scene motion has arbitrary accuracy and is oblivious to the pixel grid structure resulting from spatial sampling at the image acquisition stage i.e. by CCD arrays or other A/D post-acquisition operations. Theoretical and experimental analyses, such as the work in [21], have invented that sub-pixel accuracy has a significant impact onmotion compensated prediction error performance for a broad area of naturally moving scenes. As a result, recent standards in video compression have accepted the principle of sub-pixel accuracy for motion estimation and motion compensated prediction. Most of the sub-pel motion estimation techniques for image registration are based on interpolation. The accuracy of these methods is different for different methods and is largely determined by the characteristics of the interpolation scheme used in each case.

4.2 Fundamentals of Sub-pixel Motion estimation:

Various video processing functions gets profit from motion analysis. Inpredictive coding motion vectors are used to reduce the temporal redundancy. While in video format up-conversion the information of motion is used to interpolate the picture data at spatial positions as well as temporal positions that were not stored or transmitted. Especially in the temporal case true motion vector are required. In the application where quality is a major demand, in those cases the motion vectors are estimated using sub-pixel accuracy [17].

In the motion estimation part of the video input, interpolation in the vertical direction due to sub-sampling is not necessary. So here the Nyquist criteria are generally not met which causes alias. So researchers have developed the method where the interlacing is used either prior to motion estimation or in order to reduce the impact of alias. For an example, the recursive method [2] for the first method and generalized sampling theorem for second method. In both the cases, different sub-pixel motion vectors required different filter. The filter is dependent on the sub-pixel fraction and the interpolation used. Due to individual difference of sub-pixel fraction dependent filter, in some of the filter due to amplitude and phase characteristics unintentional preferences for certain sub-pixel fraction may occur. As a result inaccurate motion vectors are obtained which leads to sub-optimal output of the video processing algorithms which is more accurate than those motion vectors.

This implies that the demands for the interpolator used in motion estimation may be different from the demands for the interpolation used in de-interlacing or frame-rate conversion. As our main aim is to estimate true motion in a scene or a frame. It is important that the interpolation used in the motion estimation does not lead to preferred vectors. This is a strong desire that may conflict with the demand to optimally approximate the higher resolution image.

To make the process of finding motion vector using sub-pixel accuracy fully effective, sub-pixel interpolation is usually required in both spatial dimension. Both separable and non-separable interpolators can be used.

In modern video coding standards, for example H.264, both integer-pixel and fractional-pixel motion estimation (ME) is implemented. Many fast integer-pixel motion estimation algorithms have been proposed to decrease the computational complexity of integer-pixel motion estimation. With these advancements, fractional-pixel ME becomes the new bottleneck in the implementation of video encoders. Due to the large search area, the computational complexity of integer-pixel ME is extremely high and it could take 50-70% of the total encoding time. For a ± 32 search range, if full search (FS) is used there will be 4,225 candidate search positions. The number of candidate search positions for fractional-pixel ME is much smaller [18].

For example, the search range for 1/4-pixel (or quarter-pixel) fractional-pixel motion estimation is ± 3 quarter-pixels and there are only 48 candidate search positions which is discussed below. Therefore, many fast integer-pixel motion estimation algorithms are developed, e.g. three-step search [8], new three-step search [9], four-step search [12], diamond search [13], hexagon-based search [14].These algorithms reduce the number of integer pixel search points to around ten. As a consequence, considering the loads of both distortion calculation and interpolation at the

candidate positions, the computation complexity of fractional-pixel ME becomes relatively higher if no fast algorithm applied. Therefore, fast and robust fractional- pixel ME algorithm is highly desirable.

4.2 HFPS method in H.264:

ME process can be divided into two steps: the integer pixel ME and fractional pixel ME. The search range of fractional pel motion estimation in H.264/AVC JVT reference software is fixed to + 3 for quarter pel accuracy case so in many cases people prefer to choose FS at this stage for simplicity[17]. Generally integer pixel ME takes most of the computational cost of the whole ME. However with the development of fast ME algorithms the computational cost of integer pixel ME has been greatly reduced. The fractional ME has a strong impact on peak signal to noise ratio (PSNR), (about 2-3 dB significant improvement), and has also high computational complexity compared to integer ME due to complex sub-pel interpolation process[21]. Therefore the computational cost of fractional pel ME becomes comparable to that of integer pel ME, as it requires 49 points in the full fractional search method.

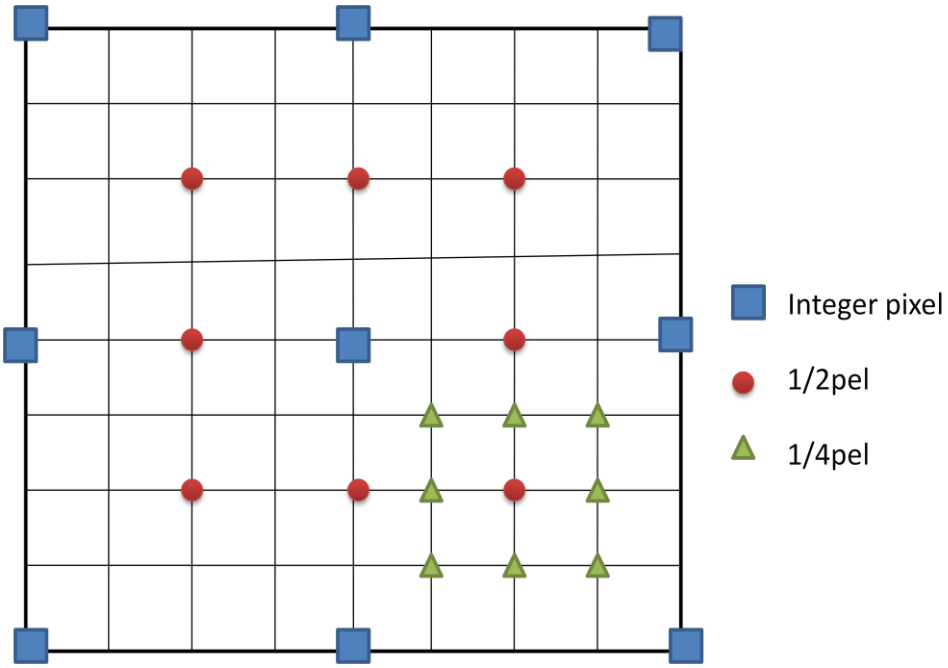


Figure 4.1: HFPS pattern

The conventional Hierarchical Fractional Pixel Search (HFPS) [17] algorithm that has been adopted in the reference software at $1/4$ pel accuracy needs to check 17 search points. Hence reducing the computational complexity for fractional pel motion search is both necessary and significant. Figure 3.1 shows the conventional hierarchical fractional pel search (HFPS) method in H.264. Initially it examines eight $1/2$ pel positions surrounding the best integer pixel position

and obtains the best 1/2 pel MV. Then it checks eight 1/4 pel positions to obtain the best 1/4pel MV.As Figure 4.1 shows, for 1/4-pel case, the search range is three 1/4-pel units. In the searching of these fractional pel positions, a 6 tap filter is used to produce the 1/2-pel positions, 1/4-pel positions is produced by linear interpolation [17]. Figure 4.1 shows the typical Hierarchical Fractional Pel Search algorithm used in JM test model. The HFPS is described by the following 3 steps:

Step-1: Check the eight 1/2-pel positions around the best integer pel position in order to findthe best 1/2-pel motion vector.

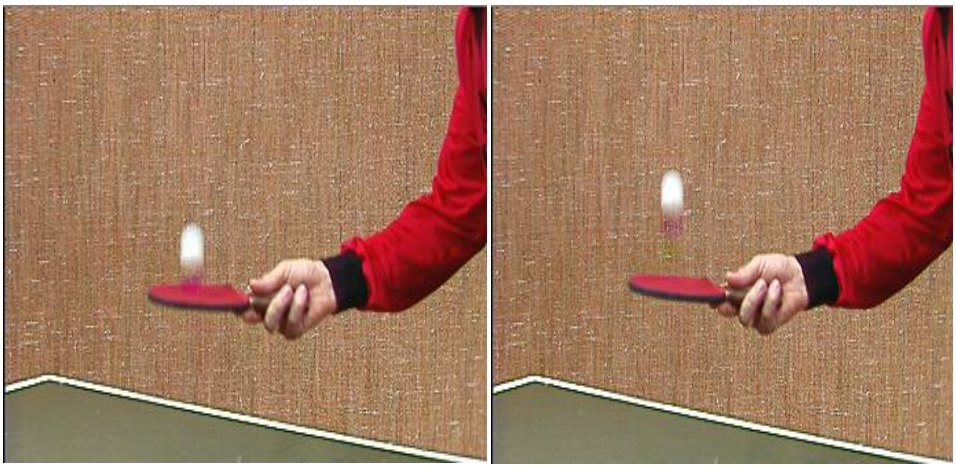
Step-2: Check the eight 1/4-pel positions around the best 1/2-pel position in order to find the best 1/4-pel motion vector.

Step-3: Select the motion vector and block-size pattern, which produces the lowest rate-distortion cost.

4.3 Study of Fractional Pixel Accuracy:

To analyze the motion-compensated error image at different fractional pixel accuracy we have taken two successive frames of some test sequence and analyzed. The motion is estimated to integer pel, half pel, quarter pel and 1/8 pel. The motion vector and motion compensated prediction error image is generated using sum of absolute difference (SAD) metric and full search procedure.

For an example we have taken the table tennis sequence to visualize the different accuracy.



(a) Reference frame (b) Current frame

Figure 4.2: Two sequence of Table tennis

The motion vectors as shown in Figure 4.3 are at the single pixel accuracy, half-pel accuracy, quarter-pel accuracy and 1/8-pel accuracy. The motion compensated error images are shown in Figure 4.4 in the above accuracy.

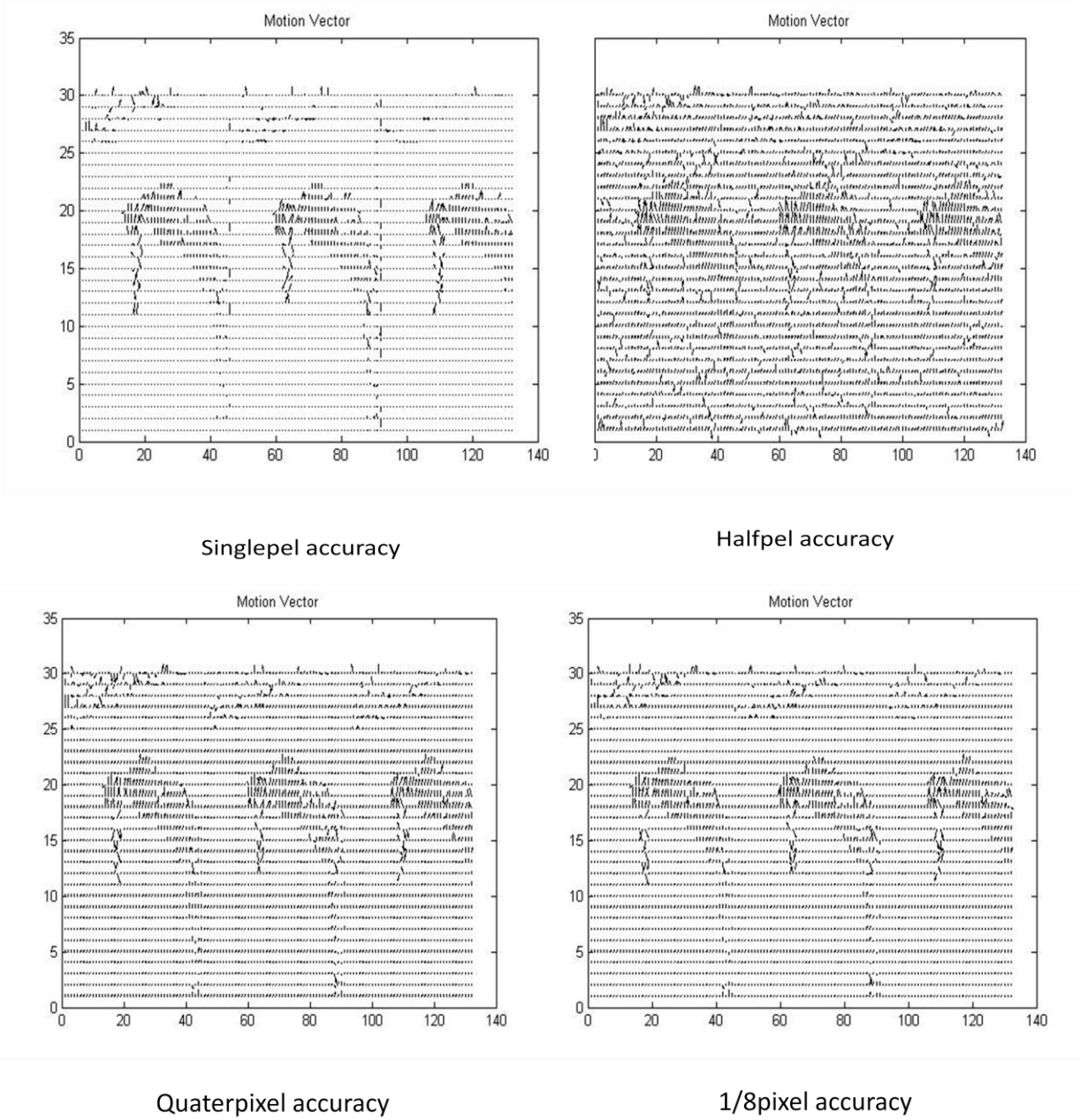


Figure 4.3: Motion vectors at different accuracy

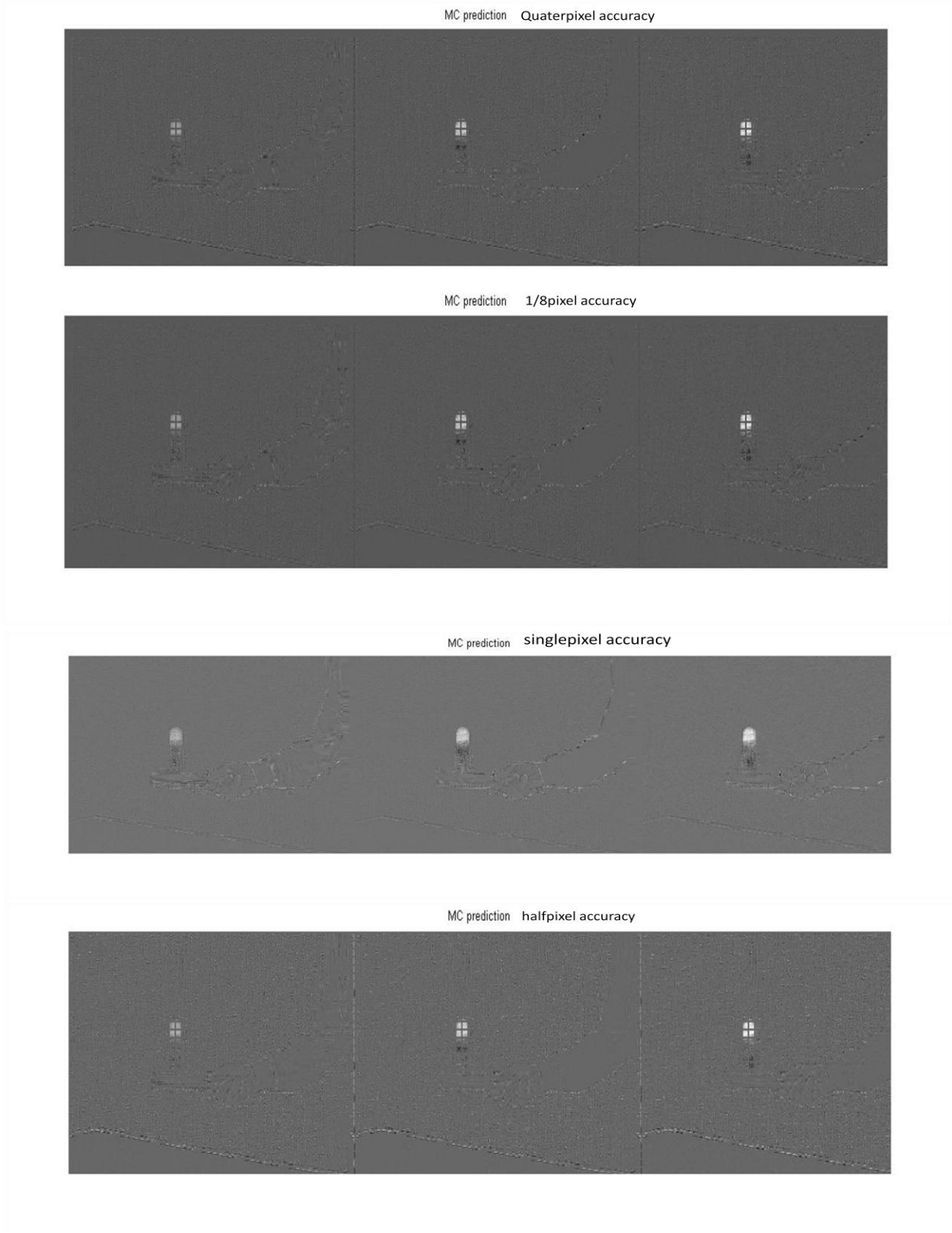


Figure 4.4: Motion Compensated error image at different accuracy

As visually it is difficult to study the changes, so we have tabulated the values for the above table tennis sequence and including that two more sequences has taken. Those are Foreman sequence and Akiyo sequence. The mean square error performance was calculated with motion compensation prediction and without motion compensation at different block sizes like 4X4, 8X8 and 12X12 at window size 8, 16 and 24 respectively.

The tables are given below:

Precisions	N=4 W=8 MSE with MC	N=8 W=16 MSE with MC	N=12 W=24 MSE with MC	MSE without MC
Singlepel	89.639	88.760	101.644	301.581
Halfpel	41.085	133.989	177.330	301.581
Quaterpel	27.742	72.346	93.919	301.581
1/8pel	20.030	50.081	68.400	301.581

Table 4.1: Study of different fractional pixel for Table tennis sequence

Precisions	N=4 W=8 MSE with MC	N=8 W=16 MSE with MC	N=12 W=24 MSE with MC	MSE without MC
Singlepel	300.708	182.553	161.831	1141.576
Halfpel	64.563	88.932	105.528	1141.576
Quaterpel	63.295	82.946	95.029	1141.576
1/8pel	62.994	82.146	93.788	1141.576

Table 4.2: Study of different fractional pixel for Foreman sequence

Precisions	N=4 W=8 MSE with MC	N=8 W=16 MSE with MC	N=12 W=24 MSE with MC	MSE without MC
Singlepel	20.247	26.071	34.006	96.136
Halfpel	4.122	13.036	21.564	96.136
Quaterpel	3.593	12.188	19.876	96.136
1/8pel	3.240	11.764	19.591	96.136

Table 4.3: Study of different fractional pixel for Akiyo sequence

4.5 Observations:

From the above tabulations, it is shown that with reduction of fractional pixel accuracy the mean square error is decreasing. As the block size decreases the value of MSE is also decreasing. So as the value of fractional pixel is reduced the more accurate picture we can get. But the major disadvantage is the computational complexity. The computational complexity is increasing as the value of the accuracy decreasing because of the interpolation.

So further we have studied how to get these fractional pixel accuracies with lower computational complexity. For that we have studied various mathematical models of mean square motion compensated (MC) prediction error in compressing moving pictures. Unlike conventional hierarchical motion estimation technique the proposed method avoid sub-pixel interpolation and subsequent search after using integer-pixel motion estimation.

CHAPTER-5

Fast Sub-pel motion Estimation technique with lower computational complexity

5.1 Introduction:

In video processing the process of ME (Motion Estimation) and MC (Motion Compensation) arerequired to reduce temporal redundancy to achieve efficient compression.ME is generally carried out using BMA (Block Matching Algorithm), spatio-temporal constraint method [5], etc. ME and MC are used to determine MVs (motion vectors) andprediction errors between the reference frame and the current frame to be coded. Since the motion estimation at integer pixel is less accurate as compared to the sub-pixel accuracy, motion estimation is often performed at sub-pixelaccuracylevelforhighercompressionefficiency.

In conventional hierarchical fractional pixel motion estimation, ME is first performed atinteger pixel level and after getting the minimum value half pixel level search is applied ateight half-pixel positions around the MV obtained[17]. This second process of searching requires half-pixel interpolation, which is a time consuming part.

In recent international standards such as H.264/AVC, the half-pixel ME is further improved to 1/4-pixel accuracy [18].There have been various architectural attempts to implement real-time video encoders. But they have the limitations in performance due to high complexity of motion estimation. So, a lot of efforts have been made to reduce time for motion estimation as it is a major portion of encoder complexity.

Half pixels positions are obtained by interpolating integer pixels, which increases computationalcomplexity. There are half-pixel accuracy searching methods [20], which is used to decrease complexity arises by the interpolation. But these methods caused to large errors due to simplified MC-error models. Here a fast sub-pixel ME techniquehaving lower computational complexity is proposed. This method is based on mathematical models of the mean-squareMC prediction errors in compressing moving pictures. Unlikeconventional hierarchical ME techniques, the proposedmethods avoid sub-pixel interpolation and following secondary search after the integer-pixel accuracy, which is useful in reducing computational time. In order to decide the coefficients of the models, the MC prediction errors of the neighboringpixels around the integer-pixel MV are taken into account. The prediction errors were already obtained during the pixel accuracy. Once the coefficients are determined, themodels are used to estimate motion

compensation prediction errors at fractional-pixel locationsaround the integer-pixel motion vector, resulting the sub-pixel MV.

5.2 Mathematical Models:

Here two mathematical models of mean square motion compensation prediction errors are described [23]. Using those models and integer-pixel accuracy, the sub-pixel accuracy are determined by evaluating the values of the model at sub-pixel position around the integer-pixel accuracy. This proposed method does not require sub-pixel interpolation or further secondary search [23].

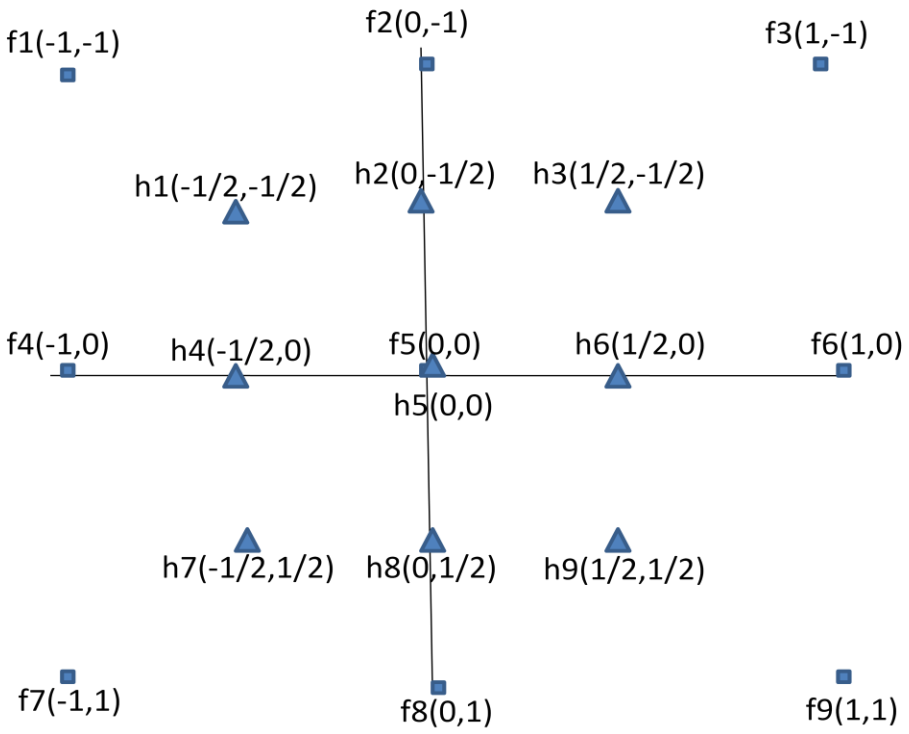


Figure 5.1: Position of Integer pixels and Half-pels

5.2.1 First Model:

In Fig 5.1, ■denotes the integer-pixel accuracy position and ▲ denotes the half-pel position surrounding the integer-pixel. Let the origin (0, 0) denotes the position of integer pixel accuracy motion vector determined through full search. 8 half-pixel positions are determined around (0, 0) or the origin (0, 0) depending on motion compensation prediction error at those 9 positions. The equation for the motion compensation prediction error plane is given below,

$$f(x,y) = c_1x^2y^2+c_2x^2y+c_3x^2y+c_4xy^2+c_5xy+c_6x+c_7y^2+c_8y+c_9 \quad (5.1)$$

The coefficients of the above equation can be determined using the motion compensated prediction errors at 3X3 integer-pixel position surrounding (0, 0). It is assumed that the mean square motion compensated prediction error at (0, 0) and neighboring integer-pixel position are already known. These 9 error values can determine 9 coefficients in equation (5.1). Substituting these 9 MC prediction errors sequentially from the upper left position (-1,-1) to the lower right position (1, 1). It is assumed that fractional pixel MC prediction error surface around the best integer pixel position, is a typical smooth unimodal surface. Therefore based on the assumption of unimodal surface, MC prediction error surface is modeled as

$$\begin{bmatrix} f(-1,-1) \\ f(0,-1) \\ f(1,-1) \\ f(-1,0) \\ f(0,0) \\ f(1,0) \\ f(-1,1) \\ f(0,1) \\ f(1,1) \end{bmatrix} = \begin{bmatrix} 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 1 \\ 1 & -1 & 1 & 1 & -1 & 1 & 1 & -1 & 1 \\ 0 & 0 & 1 & 1 & 0 & -1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \\ c_6 \\ c_7 \\ c_8 \\ c_9 \end{bmatrix} \quad (\mathbf{5.2})$$

Where f_1, f2. . . . .f_9 denotes the motion compensation errors which are known prior. The coefficients are obtained by taking inverse of equation (2), which will give,

$$\begin{bmatrix} C_1 \\ C_2 \\ C_3 \\ C_4 \\ C_5 \\ C_6 \\ C_7 \\ C_8 \\ C_9 \end{bmatrix} = \begin{bmatrix} 0.25 & -0.50 & 0.25 & -0.50 & 1 & -0.50 & 0.25 & -0.50 & 0.25 \\ -0.25 & 0.50 & -0.25 & 0 & 0 & 0 & 0.25 & -0.50 & 0.25 \\ 0 & 0 & 0 & 0.50 & -1 & 0.50 & 0 & 0 & 0 \\ -0.25 & 0 & 0.25 & 0.50 & 0 & -0.50 & -0.25 & 0 & 0.25 \\ 0.25 & 0 & -0.25 & 0 & 0 & 0 & -0.25 & 0 & 0.25 \\ 0 & 0 & 0 & -0.50 & 0 & 0.50 & 0 & 0 & 0 \\ 0 & 0.50 & 0 & 0 & -1 & 0 & 0 & 0.50 & 0 \\ 0 & -0.50 & 0 & 0 & 0 & 0 & 0 & 0.50 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} * \begin{bmatrix} f(-1,-1) \\ f(0,-1) \\ f(1,-1) \\ f(-1,0) \\ f(0,0) \\ f(1,0) \\ f(-1,1) \\ f(0,1) \\ f(1,1) \end{bmatrix} \quad (5.3)$$

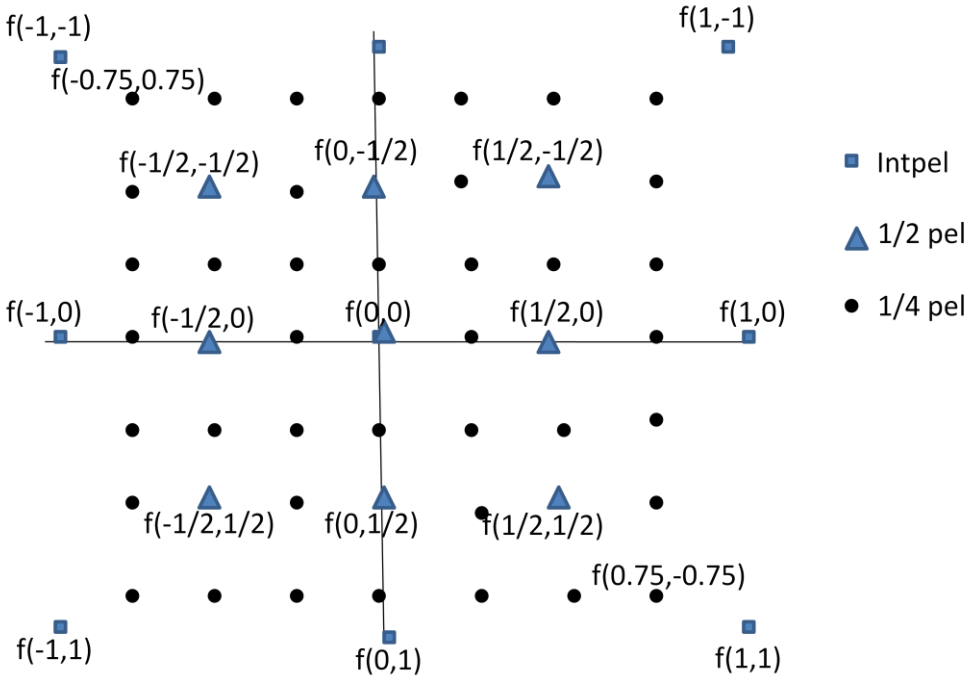
The equation (5.3) is fixed for every macroblock's sub-pixel motion estimation, no repeated calculation is required. For half-pixel accuracy putting the values -0.5, 0 and 0.5 for x and y according to the position in equation (5.1), the resultant matrix will be,

$$\begin{bmatrix} f(-0.5,-0.5) \\ f(0,-0.5) \\ f(0.5,-0.5) \\ f(-0.5,0) \\ f(0,0) \\ f(0.5,0) \\ f(-0.5,0.5) \\ f(0,0.5) \\ f(0.5,0.5) \end{bmatrix} = \begin{bmatrix} 0.0625 & -0.125 & 0.25 & -0.125 & 0.25 & -0.50 & 0.25 & -0.50 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.25 & -0.50 & 1 \\ 0.0625 & -0.125 & 0.25 & 0.125 & -0.25 & 0.50 & 0.25 & -0.50 & 1 \\ 0 & 0 & 0.25 & 0 & 0 & -0.50 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0.25 & 0 & 0 & 0.50 & 0 & 0 & 1 \\ 0.0625 & 0.125 & 0.25 & -0.125 & -0.25 & -0.50 & 0.25 & 0.50 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.25 & 0.50 & 1 \\ 0.0625 & 0.125 & 0.25 & 0.125 & 0.25 & 0.50 & 0.25 & 0.50 & 1 \end{bmatrix} * \begin{bmatrix} C_1 \\ C_2 \\ C_3 \\ C_4 \\ C_5 \\ C_6 \\ C_7 \\ C_8 \\ C_9 \end{bmatrix} \quad (5.4)$$

Similarly we can get the same type of matrix for 1/4 –pixel accuracy.

5.2.2 Implementation to H.264:

The above model is implemented in H.264/AVC. In Fig 5.2 ■ represents integer pixel, ▲ represents half pixel and ● represents quarter pixel position.



.

Figure 5.2: Position of integer pixel, half pixel and quarter pixel

Substituting the points from f(-0.75,0.75) to f(0.75,-0.75) in equation (5.1) the resultant matrix will be,

$$\begin{bmatrix} f(-0.75,0.75) \\ f(-0.5,0.75) \\ f(-0.25,0.75) \\ f(0,0.75) \\ . \\ . \\ . \\ . \\ f(0,-0.75) \\ f(0.25,-0.75) \\ f(0.5,-0.75) \\ f(0.75,-0.75) \end{bmatrix} = \begin{bmatrix} -6 & -4 & 1 & 29 & 19 & -4 & 43 & 129 & -22 \\ -4 & -7 & 1 & 16 & 33 & -5 & 25 & 149 & -15 \\ -1 & -9 & 1 & 7 & 41 & -4 & 10 & 162 & -8 \\ 0 & -9 & 0 & 0 & 44 & 0 & 0 & 166 & 0 \\ . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . \\ 0 & 66 & 0 & 0 & 44 & 0 & 0 & 91 & 0 \\ -6 & 62 & 10 & -4 & 41 & 7 & 1 & 91 & -3 \\ -8 & 49 & 25 & -5 & 33 & 16 & 1 & 93 & -11 \\ -6 & 29 & 43 & -4 & 19 & 29 & 1 & 96 & -22 \end{bmatrix} * \begin{bmatrix} f(-1,-1) \\ f(0,-1) \\ f(1,-1) \\ f(-1,0) \\ f(0,0) \\ f(1,0) \\ f(-1,1) \\ f(0,1) \\ f(1,1) \end{bmatrix} \quad (5.5)$$

By using the above matrix in equation(5.5),the position that gives the minimum motion compensation prediction error will be quarter pixel accuracy motion vector. In the 49X9 coefficient matrix an integer transform without any precision loss is used so that all elements are integers. Hence no multiplication and float point calculation are required. In addition to that the matrix has sparse features which give favorable contribution to lower complexity.

5.2.3 Second Model:

A simplified model can be obtained by eliminating the complicated higher-order terms such as x^2y^2, x^2y and xy^2. The equation (5.1) will be,

$$f(x,y)=c_1x^2+c_2xy+c_3y^2+c_4x+c_5y+c_6 \quad (5.6)$$

Here there are 6 unknown coefficients and 9 known values. Substituting the 9 MC prediction errors from the upper left position (-1,-1) to the lower right position (1, 1) into equation (5.6). We will get the matrix as follows,

$$\begin{bmatrix} f(-1,-1) \\ f(0,1) \\ f(1,-1) \\ f(-1,0) \\ f(0,0) \\ f(1,0) \\ f(-1,1) \\ f(0,1) \\ f(1,1) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & -1 & -1 & 1 \\ 0 & 0 & 1 & 0 & -1 & 1 \\ 1 & -1 & 1 & 1 & -1 & 1 \\ 1 & 0 & 0 & -1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & -1 & 1 & -1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} * \begin{bmatrix} C_1 \\ C_2 \\ C_3 \\ C_4 \\ C_5 \\ C_6 \\ C_7 \\ C_8 \\ C_9 \end{bmatrix} \quad (5.7)$$

The above equation is a typical least square problem and can be solved as

$$A.C = f$$

$$C = (A^T A).A^T.f \tag{5.8}$$

$$\begin{bmatrix} C_1 \\ C_2 \\ C_3 \\ C_4 \\ C_5 \\ C_6 \end{bmatrix} = \begin{bmatrix} 1/6 & -1/3 & 1/6 & 1/6 & -1/3 & 1/6 & 1/6 & -1/3 & 1/6 \\ 1/4 & 0 & -1/4 & 0 & 0 & 0 & -1/4 & 0 & 1/4 \\ 1/6 & 1/6 & 1/6 & -1/3 & -1/3 & -1/3 & 1/6 & 1/6 & 1/6 \\ -1/6 & 0 & 1/6 & -1/6 & 0 & 1/6 & -1/6 & 0 & 1/6 \\ -1/6 & -1/6 & -1/6 & 0 & 0 & 0 & 1/6 & 1/6 & 1/6 \\ -1/9 & 2/9 & -1/9 & 2/9 & 5/9 & 2/9 & -1/9 & 2/9 & 1/9 \end{bmatrix} * \begin{bmatrix} f(-1,-1) \\ f(0,-1) \\ f(1,-1) \\ f(-1,0) \\ f(0,0) \\ f(1,0) \\ f(-1,1) \\ f(0,1) \\ f(1,1) \end{bmatrix} \tag{5.9}$$

For half-pixel accuracy putting the values -0.5, 0 and 0.5 for x and y according to the position in equation (5.6), the resultant matrix will be,

$$\begin{bmatrix} f(-0.5,-0.5) \\ f(0,-0.5) \\ f(0.5,-0.5) \\ f(-0.5,0) \\ f(0,0) \\ f(0.5,0) \\ f(-0.5,0.5) \\ f(0,0.5) \\ f(0.5,0.5) \end{bmatrix} = \begin{bmatrix} 1/4 & 1/4 & 1/4 & -1/2 & -1/2 & 1 \\ 0 & 0 & -1/4 & 0 & -1/2 & 1 \\ 1/4 & -1/4 & 1/4 & 1/2 & -1/2 & 1 \\ 1/4 & 0 & 0 & -1/2 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 1/4 & 0 & 0 & 1/2 & 0 & 1 \\ 1/4 & -1/4 & 1/4 & -1/2 & 1/2 & 1 \\ 0 & 0 & 1/4 & 0 & 1/2 & 1 \\ 1/4 & 1/4 & 1/4 & 1/2 & 1/2 & 1 \end{bmatrix} * \begin{bmatrix} C_1 \\ C_2 \\ C_3 \\ C_4 \\ C_5 \\ C_6 \end{bmatrix} \tag{5.10}$$

For the same position of single pixel, half pixel and quarter pixel as shown in the Figure 5.2,Substituting the points from f(-0.75,0.75) to f(0.75,-0.75) in equation (5.6) the resultant matrix will be,

$$\begin{bmatrix} f(-0.75,0.75) \\ f(-0.5,0.75) \\ f(-0.25,0.75) \\ f(0,0.75) \\ . \\ . \\ . \\ . \\ f(0,-0.75) \\ f(0.25,-0.75) \\ f(0.5,-0.75) \\ f(0.75,-0.75) \end{bmatrix} = \begin{bmatrix} -6 & 0 & -3 & 25 & 18 & 0 & 47 & 25 & 16 \\ -11 & 11 & -9 & 16 & 28 & -1 & 33 & 36 & 19 \\ -14 & 17 & -13 & 9 & 35 & 0 & 21 & 42 & 25 \\ -14 & 19 & -14 & 3 & 37 & 3 & 11 & 44 & 33 \\ . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . \\ 11 & 44 & 11 & 3 & 37 & 3 & -14 & 19 & 8 \\ 3 & 42 & 21 & 0 & 35 & 9 & -13 & 17 & 9 \\ -3 & 36 & 33 & -1 & 28 & 16 & -9 & 11 & 11 \\ -6 & 25 & 47 & 0 & 18 & 25 & -3 & 0 & 16 \end{bmatrix} * \begin{bmatrix} f(-1,-1) \\ f(0,-1) \\ f(1,-1) \\ f(-1,0) \\ f(0,0) \\ f(1,0) \\ f(-1,1) \\ f(0,1) \\ f(1,1) \end{bmatrix} \quad (5.11)$$

5.3 Results and Analysis:

To analyze the above two models we have taken five different sequences named as (1) Akiyo sequence (2) Foreman sequence (3) Table tennis sequence (4) salesman sequence and (5) Caltrain sequence. Among the above sequences (4) and (5) are gray scale images and rests are RGB images. From all the sequences two frames are taken one as current frame and other as reference frame.Using the above proposed mathematical models we have calculated the computational time and mean square error with motion compensation at different macroblock size such as 32, 16 and 8.

SEQUENCES	Model-1 MSE with MC			Model-2 MSE with MC		
	MbSize=32	MbSize=16	MbSize=8	MbSize=32	MbSize=16	MbSize=8
Akiyo	933.77	632.58	345.31	43.35	42.46	33.81
Foreman	507.65	457.17	355.67	332.34	158.80	58.71
Salesman	25.63	18.85	8.99	18.88	10.16	7.55
Caltrain	1238.18	1108.85	896.07	184.32	146.32	98.25
Table tennis	985.78	650.56	269.83	684.90	317.38	255.44

Table 5.1: Comparison study of MSE with MC for Model-1 and Model

In the above table we have compared the performance of the two mathematical models by taking mean square error with motion compensation. Here we observed that the value of MSE is decreasing with reduced macroblock size. The MSE performance is better in model-2 as compared to model-1. Thus the quality will be improved in model-2. Because the value of MSE with MC is less in overall sequences of model-2.

Motion Estimation Time (Sec)		
SEQUENCES	Model-1	Model-2
Akiyo	0.374	0.296
Foreman	0.296	0.149
Salesman	0.109	0.109
Caltrain	0.109	0.124
Table tennis	0.327	0.312

Table 5.2: Comparison study of Model-1, Model-2 and HFPS by measuring ME time (sec.)

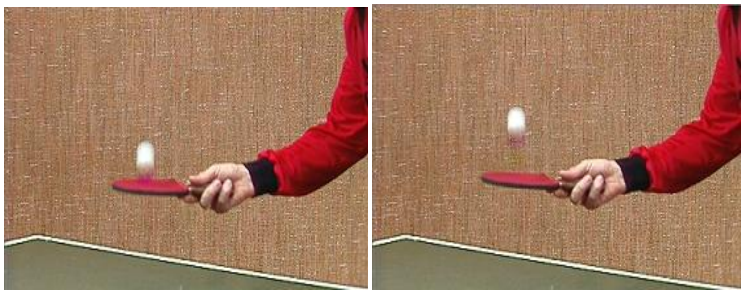
The above table compares the fractional pixel motion estimation time of both the model. From the tabulation it is clear that the computational time is greatly reduced in both the models. But there is not much difference in motion estimation time between the model-1 and model-2 as the size of the matrix is same in both the cases. So it is cleared that the computational complexity is decreasing with the help of these two models.

In conventional quarter-pixel ME algorithm, the quarter-pixel ME is conducted into two stages: first, the eight 1/2-pixel positions are obtained surrounding the best integer-pixel position in order to find the best 1/2-pixel motion vector; Second, the eight 1/4-pixel positions are obtained surrounding the best 1/2-pixel position in order to find the best 1/4-pixel motion vector. In both the stages, cost function and interpolation need to be calculated. Using proposed algorithm, only one step is required to get quarter-pixel MV. So the proposed algorithm avoidsusing cost function and interpolation which require a lot of computation time in fractional-pixel ME.

(a)Two frames of Akiyo Sequences

(b)Two frames of Caltrain Sequences

(c)Two frames of Foreman Sequence



(d)Two frames of Table tennis

(e)Two frames of Salesman Sequence

Figure 5.3: Five different sequences used in the models

5.4 Conclusion:

The proposed algorithm is carried out on MATLAB 2007. Five widely used test sequences i.e. Akiyo, Foreman, salesman, Caltrain and Table tennis are used in this experiment. The number of encoded frame is 2 for every sequence. One is reference frame and the other is current frame. The comparison criteria is chosen to be the mean square error (MSE) to evaluate quality and fractional-pixel motion estimation time to evaluate computational complexity.

From the Table 5.1 it is concluded that the model-2 is better than model-1 by quality as the value of MSE with motion compensation is less as compared to model-1 also the value of MSE is decreasing as the size of the macro-block is decreasing .

From the Table 5.2 it is concluded that in both the model the computational complexity is greatly reduced.

Conclusion & Future work:

The thesis work primarily focuses on motion estimation part of H.264/AVC as it is the most time consuming part of the encoder. Two types of motion estimation algorithms were studied. One is single-pixel motion estimation algorithm and another is fractional-pixel motion estimation algorithm.

In single-pixel algorithm we tried to improve the existing algorithms i.e. three step search (TSS) and four step search (4SS) by using an adaptive threshold technique.By using this technique the no. of search point is reduced in both the existing algorithms. In case of improved TSS the search point is reduced by 30%-40% as compared to conventional three step search TSS. In case of improved four step search the search point is reduced by 50%-70% as compared to conventional four step search (4SS) which in turn reduces the computational complexity and also the performance is compared to other block matching techniques i.e. full search (FS) and new three step search (NTSS) and found better compared to these algorithms in terms of reduced no. of search point per macro-block and reduced motion estimation time. But the quality of the improved algorithms remains same as compared to the existing three step search (TSS) and four step search (4SS). So in future we can work out to improve the quality.

In fractional-pixel motion estimation algorithm we have studied various fractional pixels like half-pixel, quarter-pixel, 1/8-pixel accuracy and found 1/8-pixel accuracy is better than the half-pixel and quarter-pixel by quality and we concluded that as the value of fractional-pixel decreases the quality of the picture is improved but due to interpolation the computational complexity increases which is again a disadvantage. In the conventional HFPS (Hierarchical Fractional Pixel search) and the proposed algorithm work is extended up to quarter-pixel accuracy. So in future to improve the quality the 1/8-pixel accuracy or pixel accuracy less than that may be used by keeping the computational complexity as much less as possible.

REFERENCES:

[1] Puri, X. Chen, and A. Luthra, "Video coding using the H.264/MPEG-4 AVC compression standard," Signal processing*:* Imagecommunication, vol. 19, pp. 793-849, 2004.

[2] R. Schafer, T.Wiegand, H. Schwarz, "The emerging H.264/AVC standard," EBU Technical Review, January 2003

[3] K. Rijkse, "H.263:Video coding for low-bit-rate communication, "IEEE CommunicationsMagazine, vol. 34, pp. 42-45, 1996.

[4] L.Vander, L.Cuvelier,B.Maison,P.Quelez and P.Delogue, "Motion-Compensated conversion from interlaced to progressive formats." Signal Processing: Image Communication (6), pp.193-211,1994.

[5] K.S. Thyagarajan,'' Still image and video compression with MATLAB", pp.42-68, November 2010.

[6] M. J. Chen, L. G. Chen, and T. D. Chiueh, "One-dimensional full search motion estimation algorithm for video coding," IEEE Trans. Circuits Syst. Video Technol., vol. 4, pp. 504–509, Oct. 1994.

[7] Richardson, H.264 and MPEG-4 Video Compression: Video Coding for Next Generation Multimedia: Wiley, 2003.

[8] Xuang Jing, Chau, L.-P, "An Efficient Three Step search Algorithmfor Block Motion Estimation," IEEE Trans. Multimedia, Vol:6,pages.435-438,2004.

[9] Renxiang Li, Bing Zeng, and Ming L. Liou, "A New Three-Step Search Algorithm for Block Motion Estimation", IEEETrans. Circuits AndSystems For Video Technology, vol 4., no. 4, pp. 438-442, August1994.

[10] Jianhua Lu, and Ming L. Liou, "A Simple and Efficent Search Algorithm for Block-Matching Motion Estimation", IEEETrans.Circuits And Systems For Video Technology, vol 7, no. 2, pp. 429-433, April 1997.

[11] L.-K. Liu and E. Feig, "A block-based gradient descent search algorithm for block motion estimation in video coding," IEEE Trans. Circuits Syst. Video Technol., vol. 6, no. 4, pp. 419–422, Aug. 1996.

[12] Lai-Man Po, and Wing-Chung Ma, "A Novel Four-Step Search Algorithm for Fast Block Motion Estimation", IEEETrans. CircuitsAnd Systems For Video Technology, vol 6, no. 3, pp. 313-317, June 1996.

[13] Shan Zhu and Kai-Kuang Ma, “Anew diamond search algorithm forfast block-matching motion estimation”, Image Processing, IEEE Transactions on,Volume: 9**,** Issue: 2**,** Feb. 2000Pages: 287 -290.

[14] Zhu, X. Lin, and L.-P. Chau, “Hexagon-based search pattern for fast block motion estimation,” IEEE Trans. Circuits Syst. Video Technol., vol. 12, no. 5, pp. 349–355, May 2002

[15] W. Lam, L. M. Po and C. H. Cheung, "A New Cross-Diamond Search Algorithm for Fast Block Matching Motion Estimation", Proceeding of 2003 IEEE International Conference on Neural Networks and Signal Processing, pp. 1262-1265, Dec. 2003, Nanjing, China

[16] Yao Nie, and Kai-Kuang Ma, “Adaptive Rood Pattern Search for Fast Block-Matching Motion Estimation”, IEEE Trans. Image Processing, vol 11, no. 12, pp. 1442-1448, December 2002.

[17] Z. Chen, Y. He, and Y. Chen, "Fast Integer and fractional pel motion estimation," Joint Video Team (JVT) of ISO/IEC MPEG and ITU-TVCEG, JVT- F, 2002.

[18] HumairaNisar,Tae-Sun Choi, “Fast and efficient fractional pixel motion estimation for H.264/AVC video coding”,978-1-4244-5654-3/09, 2009 IEEE.

[19] ArohBarjatya, Student member IEEE, “Block-Matching Algorithm for Motion Estimation”, DIP 6620 Spring 2004 final project paper.

[20] Hung-Min,Chen,Po-Hung,Chen,Ching-Chung Liu,Jhong-Kai Lin, “A Modified Directional Gradient Descent Search for Fast Block Motion Estimation”,International Symposium on computer ,consumer and control,978-0-7695-4655-1/12,2012 IEEE.

[21] Fei Wang,YuanLi,Huizhu Jia, Xiandong Xie,Wen Gao, “An efficient fractional motion estimation architecture for AVS real-time full HD video encoder”,978-1-4577-1775-8/12,2012 IEEE.

[22] Z. Chen, J. Xu, Y. He, and J. Zheng, “Fast integer-pel and fractional pel motion estimation for H.264/AVC,” Journal of Visual Communication And Image Representation,vol.17,pp.264-290,2006.

[23] J.W.Suh, J. Jeong, “Fast Sub-pixel Motion Estimation Techniques Having Lower Computation Complexity,” IEEE transaction on consumer and electronics, vol. 50, pp. 968-973, Aug. 2004.